Multimedia Contents

33. Visual Object Class Recognition

Michael Stark, Bernt Schiele, Aleš Leonardis

Object class recognition is among the most fundamental problems in computer vision and thus has been researched intensively over the years. This chapter is mostly concerned with the recognition and detection of basic level object classes such as cars, persons, chairs, or dogs. We will review the state of the art and in particular discuss the most promising methods available today.

33.1 **Object Classes** 825
33.2 **Review of the State of the Art** 826
33.2.1 Bag-of-Words Models 826
33.2.2 Spatial Pyramid Matching 828
33.2.3 Flexible Part Models 829
33.2.4 Histogram of Oriented Gradients 831
33.2.5 The Deformable Part Model 832
33.2.6 Convolutional Neural Networks 834
33.2.7 Fine-Grained Categorization 836
33.2.8 Datasets 836
33.3 **Discussion and Conclusions** 837
33.3.1 Task 837
33.3.2 Data 838
33.3.3 Efficiency 838
References 838

33.1 Object Classes

Generic object class recognition has been one of the goals of computer vision since its beginnings [33.1, 2]. In this context, it is important to emphasize that the notion and the abstraction level of object classes is far from being uniquely and clearly defined. Notably, the question of how humans organize knowledge at different levels has received much attention in cognitive psychology [33.3, 4]. Taking an example from Brown's work, a dog can not only be thought of as a *dog*, but also as a *boxer*, a *quadruped*, or in general an *animate being* [33.3]. Yet, *dog* is the term that comes to mind most easily, which is by no means accidental. Experiments show that there is a *basic level* in human categorization at which most knowledge is organized [33.5]. According to *Rosch* et al. and *Lakoff* [33.5, 6], this basic level is also:

- The highest level at which category members have similar perceived shape
- The highest level at which a single mental image can reflect the entire category
- The highest level at which a person uses similar motor actions for interacting with category members
- The level at which human subjects are fastest at identifying category members
- the first level named and understood by children.

Given that the basic level of categorization is easiest for humans, it should be a good starting point for machine vision, too [33.7]. The next lower level, *subordinate categories*, corresponds to the individual level used in object identification, e.g., the next higher level, *super-ordinate categories*, requires a higher degree of abstraction and world knowledge. The following will thus discuss state-of-the-art approaches first for basic-level categorization but then also briefly touch upon subordinate categorization for fine-grained classification.

Avidbots Ex1016
Page 859 of 2154

33.2 Review of the State of the Art

One can think of all current methods in object class recognition as being part-based methods, in which object class instances are described by a set of parts (such as human body parts, for instance), and the topology between the parts (such as the kinematic tree defined by the joints that connect the human body parts). The following sections give a systematic treatment of different instantiations of state of the art, part-based methods, ranging from the bag-of-words model (Sect. 33.2.1) and spatial pyramid matching (Sect. 33.2.2), over flexible part models (Sect. 33.2.3) and rigid histogram of oriented gradients templates (Sect. 33.2.4) to deformable part models (Sect. 33.2.5) and convolutional neural networks (Sect. 33.2.6). Table 33.1 summarizes the various part-based methods presented in this section. Later sections highlight recent trends in fine-grained categorization (Sect. 33.2.7) and datasets for visual recognition (Sect. 33.2.8).

Table 33.1 Classification of some prominent approaches as part-based models

Model	Parts	Topology
Bag-of-words	Feature clusters	No spatial information coded
Spatial pyramid	Feature clusters	Hierarchical but rigid spatial grid
Constellation model	Feature clusters	Full covariance matrix between clusters
Implicit shape model	Feature clusters	Star-shaped model
HOG	HOG-cells	Rigid spatial grid
DPM	HOG-parts	Star-model
Pictorial structures model	Semantic parts	Tree structure
Convolutional neural networks	Convolution filters	Spatial spooling and hierarchy

33.2.1 Bag-of-Words Models

Apart from allowing to establish pointwise correspondences between multiple images of the same scene, local image patches also provide the foundation for a variety of object class representations for visual recognition. Inspired by the natural language processing literature, the *bag-of-words* (BOw) or *bag of features* model [33.8] represents an image as an unordered collection (a bag) of local patches. This representation assumes that most of the important information about the image content is captured purely by which patches occur (and how often), irrespective of where they occur. While this is clearly an approximation, it seems reasonable from the text document analogy: even if the words in a text document are shuffled randomly, the mere occurrence of certain key terms provides a strong cue about the document's topic. Likewise, a collection of local image patches, such as two eye patches, mouth, and nose, provide strong cues for the presence of a face in an image.

The bag-of-words model is typically based on a codebook of visual words: a fixed, discrete set of visual features relative to which all others are described (Fig. 33.1c). An input image (Fig. 33.1a) can then be described relative to that codebook, by counting how often each codeword appears (Fig 33.1b). In the following paragraphs, we describe in detail the extraction of visual features, the generation of codebooks of visual words, and their use in image classification using probabilistic generative and discriminative models.

Feature Extraction

Locations and scales for feature extraction are typically either picked sparsely, based on an interest operator, such as the difference of Gaussian (DOG)

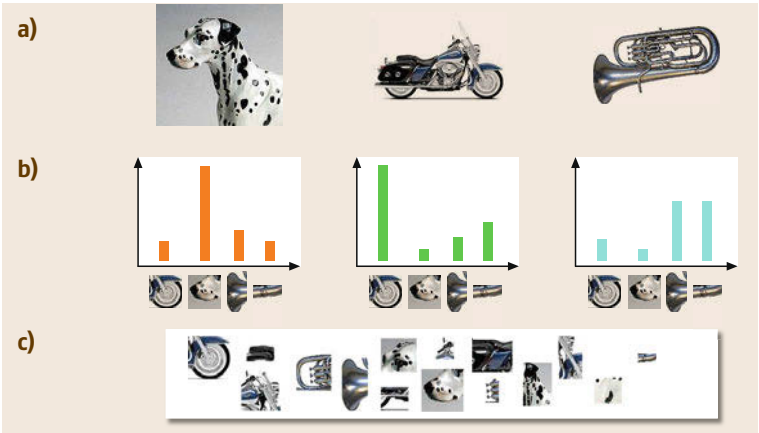


Fig.33.1a–c The bag-of-words representation. **(a)** Image; **(b)** BOw histogram; **(c)** codebook of words

Avidbots Ex1016
Page 860 of 2154

interest point detector for scale invariant feature transform (SIFT) [33.9], Harris corners, scale- and affine-invariant Hessian–Laplace interest points [33.10], maximally stable extremal regions (MSER) [33.11], or based on dense feature sampling using a fixed spatial grid [33.12]. While interest operators generally provide well-localized, repeatable features with sufficient recall, the increasing availability of compute power today has made very dense feature sampling feasible (e.g., with a stride of only one pixel, resulting in many thousands of features in an image) – in the limit, dense sampling typically outperforms sparse interest operators in image classification as interest point detectors are designed to sample particular types of image structure whereas dense sampling allows to sample any type of image structure (note that this might be different for other applications, such as structure from motion, in which it is more important to find smaller numbers of features with high precision).

Codebook Generation

Codebooks of visual features are typically constructed by applying an unsupervised clustering algorithm, such as *k*-means or hierarchical agglomerative clustering [33.13], to a training set of image feature vectors. The result is a vector-quantized representation of the visual feature space that groups similar feature descriptors into clusters. Each cluster thus effectively generalizes over its members, such that small visual differences are suppressed – this is often seen as a crucial trait for categorization even though larger codebook sizes typically result in better performance. Each cluster is identified with a representative member, i. e., its centroid (for *k*-means) or medoid (for agglomerative clustering). In order to represent a new image with respect to the generated codebook, image features are extracted, and matched to the representative of each cluster (e.g., by computing pairwise distances and defining a matching threshold). The result of this matching is then recorded (e.g., as a histogram over the number of times each cluster has been matched) to yield the image or object representation. Note that in the context of BOw models the terms object and image representation are synonymous.

For codebook generation, the number of clusters is an important free parameter: it has been found that large codebook sizes typically improve recognition performance, since they represent the original feature space more faithfully, with lower reconstruction error. On the other hand, large codebooks generalize less, and come at the cost of increased computational cost when building bag-of-words representations and classifying them with nonlinear classifiers. Typically, codebooks with several and even 10 000s (multiples of ten thousand) of clusters are chosen for optimal performance [33.14].

Generative Models

Inspired by their success in the text domain, generative probabilistic models have also been adopted for image classification in connection with BOw representations. The arguably simplest one is the naive Bayes model [33.8, 15], in which the posterior probability of an image having a certain class factors into independent contributions from each histogram bin (i. e., for a given class, knowing about the presence of one feature does not make the presence of another feature any more or less likely). While this model is agnostic of correlations that often exist between features (for instance, having already observed two wheel features in a sideview of a car makes it unlikely to observe one more, assuming that only a single car is present in the image), it often works well in practice and is appealing due to its simplicity.

In line with the text domain, more powerful models have been proposed for image classification that explicitly capture the co-occurrence of features through a generative process. Here, the features contained in an image are assumed to be generated in a sequence of steps, where each step determines the choices available for the next step. More precisely, each step is implemented by sampling from a probability distribution, conditioned on the outcome of the previous step.

A popular class of generative models is the *topic model* [33.16, 17]: an image is considered to be an orderless document of visual words, composed of multiple topics (in an image depicting a beach scene, there could be a water topic, a sand topic, and a sky topic, for instance). In order to generate a particular feature in the image, a topic is first sampled from a discrete distribution over topics (a beach scene could be a roughly uniform distribution over water, sand, and sky). Then, a visual feature is sampled from a discrete distribution over features, conditioned on the previously sampled topic (e.g., for a water topic, the sampled feature is likely to be a blue image patch with a ripple pattern). In practice, the assignment of visual features to topics is represented through latent variables, which means that it has to be inferred from other quantities that can be observed (such that the likelihood of the features is maximized). This is typically implemented by means of the expectation maximization (EM) [33.18] algorithm, which alternates (re-)estimation of topic distributions (E-step) and imputation of latent variables (M-step) in an iterative procedure.

In the literature, there exists a large variety of topic models, ranging from the basic probabilistic latent semantic analysis (PLSA) [33.16] over its Bayesian extension, latent Dirichlet allocation (LDA) [33.17], to

Avidbots Ex1016
Page 861 of 2154

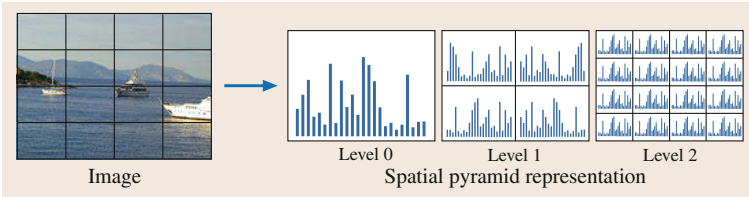


Fig. 33.2 Spatial pyramid matching as a multilevel BOw representation (after [33.20])

infinite hierarchical models of images and their features [33.19].

Discriminative Models

Once an image is represented as a histogram of matched features with respect to a codebook, it can be classified by means of any discriminative classifier that is applicable to histograms. Popular choices include support vector machines [33.21] with histogram intersection kernels or their approximate counterpart [33.22], boosting [33.23], or random forest classifiers [33.24].

Extensions

While the bag-of-words representation inherently ignores spatial information, it can be re-introduced by restricting the support of a bag-of-words representation to a local image region, and processing multiple regions independently. Specifically, instances of an object class can be localized by sliding a rectangular window over the image under consideration, extracting a bag-of-words representation at each location, and feeding the result into a classifier that distinguishes the object class of interest from a background. Objects of different sizes can be localized by applying this scheme to multiple re-scaled versions of the input image [33.25] (called an image pyramid).

Rather than applying the sliding window in a brute force fashion, it has been realized that the classifier score that an image window can attain can be bounded from above by the scores of the subwindows that it contains. This observation gives rise to an efficient subwindow search [33.26] that greatly reduces the number of classifier evaluations needed in connection with sliding window bag of words representations, making powerful but slow nonlinear kernel SVMs applicable.

33.2.2 Spatial Pyramid Matching

Instead of completely ignoring the spatial layout of local features in an image, it seems reasonable to include at least a rough notion of locality in order to increase the discriminative power of an object class representation. Spatial pyramid matching implements this notion as an extension of the bag-of-words model (Sect. 33.2.1), by extracting multiple bag-of-words representations for a single input image at different spatial locations and scales, and aggregating these local representations into a single global representation for the entire image. Specifically, spatial pyramid matching (SPM) [33.27] extracts bag-of-words histograms for each cell in a rectangular grid overlaid onto the input image (typically, coarse grids of 2^l for $l \in \{0, 1, 2\}$). This process is repeated multiple times with different numbers of cells (e.g., four by four plus two by two plus a single cell), forming an overcomplete pyramid structure of decreasing spatial resolution (Fig. 33.2). The resulting bag-of-words histograms for each cell are concatenated (and re-normalized). The appeal of SPM lies in representing the spatial layout at multiple levels of resolution without committing to any one in particular – instead, the representations of all resolutions are kept, and have the chance to contribute to image classification in case they prove useful.

The basic SPM methodology appears in numerous variations in the literature, but often follows the same pipeline of four steps (feature extraction, feature coding, spatial pooling, and classification), each of which we review in the following sections.

Feature Extraction

Like the basic bag-of-words model, the SPM model has been used in connection with various feature types, ranging from SIFT [33.9] appearance features to local edge configurations [33.28] and self-similarity features [33.29].

Feature Coding

Representing visual features relative to a pretrained codebook obtained by vector quantization is only one special case of feature coding (performing a change of representation), and several improvements have been proposed in the literature [33.30–32]. Most importantly, it has proven beneficial to explicitly formulate the codebook training and feature coding as a joint optimization problem, such that the original features can be reconstructed with small error from their respective codes. In addition, this formulation allows us to relax the assumption that each feature is matched to exactly one codebook entry by replacing the cardinality constraint with an L1 regularization. The resulting sparse coding (SC) formulation [33.31] has been shown to outperform the original SPM [33.27] by significant margins, even

Avidbots Ex1016
Page 862 of 2154

when combined with a linear SVM (while SPM works best in connection with nonlinear histogram intersection or chi-square kernels, which are computationally expensive). An improved version of SC is given by locality constrained linear coding (LLC) [33.32]. It replaces sparsity by locality constraints that encourage features to be reconstructed by nearby features in feature space, such that similar features yield similar codes (which is obviously desirable in order to facilitate classifier learning).

Instead of relying solely on counting the number of occurrences of visual features in a spatial region as promoted by SPM, SC, and LLC, the Fisher vector [33.30] method characterizes that region by how well a pretrained generative probabilistic model (a Gaussian mixture model, GMM) explains the contained features. Specifically, it builds a vector from the derivatives of the log-likelihood of the image features, given the mixture coefficients, means, and diagonal covariances of a GMM. Note that this vector has much more entries than a histogram of occurrences using the same GMM as a codebook, allowing to build highly discriminative representations already for small numbers of GMM mixture components (typically only a few hundred).

Both sparse coding and Fisher vector representations haven proven to be very effective for image classification in connection with spatial pyramids, and have been successfully applied also in large-scale settings, such as classification on ImageNet [33.33] (Sect. 33.2.8).

Spatial Pooling

Apart from the classic pyramid scheme [33.27] in which histograms of different pyramid levels are concatenated, retaining the maximum value of a particular bin over a spatial region or even multiple pyramid levels (max-pooling) has gained increasing attention [33.31], in particular in connection with CNNs (Sect. 33.2.6). Max-pooling has the advantage that it promotes invariance to local translations and scale changes, and has been found to be present in the human visual cortex (V1) [33.34].

In addition to pooling features in the two-dimensional (2-D) image plane, more object-centric pooling techniques have been developed in the context of fine-grained categorization [33.35, 36] (Sect. 33.2.7).

Classification

The last stage in an SPM pipeline is typically given by a discriminative classifier that is trained on a training set of fixed length vectors created from the SPM representations. Depending on the sophistication of the feature coding step, support vector machines with nonlinear histogram intersection or chi-square [33.27] or linear kernels [33.30–32] achieve state-of-the-art performance. Further extensions include the use of multiple kernels with learned contribution weights [33.37]. When evaluated as part of a classification cascade, multiple kernel learning (MKL) can also be used in a sliding window object class detector [33.38].

33.2.3 Flexible Part Models

While orderless bags of visual features (Sect. 33.2.1) or spatial pyramids with a fixed grid layout (Sect. 33.2.2) have proven to be effective object class representations for classification, they both fail to explicitly capture the flexible nature of physical objects that are composed of multiple parts – as an example, consider the human face with eyes, eyebrows, mouth, nose, and ears: obviously, the precise shape and spatial arrangement of these parts will vary from person to person, but variations in shape and arrangement are constrained to lie within a plausible range (i. e., the parts are not unordered as in a bag-of-words representation). It also seems intuitive that the constraints that govern the range of plausible variations should be soft, and not bound to a fixed binning or a fixed spatial grid (as in spatial pyramid matching). In this section, we review a family of flexible part models that implement this intuition.

Constellation Model

The constellation model [33.39–41] represents an object class as a flexible arrangement of a fixed number of parts. It consists of two components: first, a model of the appearance of each part, and second, a model of the relative spatial layout of the parts. It is formulated in a fully probabilistic way, such that each part is characterized by a probabilistic, generative model of its appearance, and the spatial layout of the parts is governed by a joint Gaussian distribution that captures how the location (and scale) of each part is expected to vary with the location of each other part. Interestingly, the definition of what constitutes a part is driven entirely by the training data: parts are chosen and placed such that the likelihood of the training data, given the model, is maximized during training. This can be seen as an instance of a missing data problem (the association between features and parts is unknown), and solved by means of the expectation maximization (EM) algorithm.

Unfortunately, the representative power of the joint Gaussian spatial distribution comes at a cost: the fully connected dependency structure demands examining each of the exponentially many combinations of placing all parts in an image in order to find the global optimum – the exact inference problem is known to be intractable. As a consequence, the original constellation

Avidbots Ex1016
Page 863 of 2154

model [33.40, 41] is limited to a few dozen candidates per part, per image, limiting its generality. An alternative approach to solving the exact inference problem is to resort to an approximation, e.g., by sampling likely constellations from part detection proposals [33.42]. In this way, the number of part candidates can be scaled up to several thousands.

Due to its probabilistic nature, the constellation model has also been extended to a full Bayesian model [33.43], in which the object class model itself is governed by a prior distribution over models. This abstraction allows to separate characteristics of one particular object class (e.g., the particular shape of a horse's head) from properties that are shared by classes (e.g., all quadrupeds are four legged). These shared properties can be re-used in multiple object class detectors (e.g., for horses, cows, dogs, etc.), facilitating learning from limited training data. The underlying idea of re-use of information is known as transfer learning in the machine learning literature, and has been successfully applied for image classification [33.43] as well as object class detection.

Implicit Shape Model

The implicit shape model (ISM) [33.44] relaxes the assumption of the constellation model that the locations and scales of all parts are mutually dependent. Instead, it limits the dependency structure to a star topology, in which part locations and scales are conditionally independent, given the object center (i. e., given the object center, revealing information about the location of one part does not alter the expectation of where to observe any other part). While this model clearly constitutes an approximation of the physical world, it greatly simplifies the inference problem of finding the configuration with maximum score (there exists an algorithm that is exact and efficient).

Specifically, the ISM represents an object class as a collection of local feature matches – the supply of matchable features is given by a codebook of visual words, as it is also used in bag-of-words (Sect. 33.2.1) or spatial pyramid representations (Sect. 33.2.2). In contrast to those, however, each ISM codebook entry also maintains a list of feature matches from a training set of images: each match specifies where the match occurred relative to the object center. As a result, this list of occurrences constitutes a nonparametric probability density estimate of the location of the object center, given the matched feature – in other words, each feature can cast votes for likely object center positions (e.g., a wheel feature can cast two votes: one to the left of it, assuming that it is the front wheel of a car, and one to the right, assuming it's the back wheel). This voting procedure can directly be translated into a recognition algorithm: at recognition time, all test image features cast their votes into a three-dimensional (3-D) voting space (x, y, and scale). All votes are accumulated, and since multiple features are likely to agree on the true object center, the maxima of this voting space can be considered candidate object detections (note that this voting can be considered a generalized Hough transform). In practice, continuous voting space maxima are found using a mean-shift algorithm.

In order to increase its discriminative power, the basic ISM model is extended by a verification step [33.44], in which object detection candidates are verified based on how consistent they are with respect to foreground–background segmentation. To that end, feature occurrences stored as part of the codebook are enriched with segmentation mask fragments (which have to be provided as additional supervision during training). At recognition time, the consistency of the cast segmentation mask votes is quantified based on minimum description length (MDL). In addition, it has been suggested to learn weights that regulate the contributions of feature matches to the final detection hypotheses in a discriminative way (i. e., optimized to distinguish object and background firings). This *discriminative Hough transform* [33.45] further improves the performance of the ISM and constitutes the basis for the *poselet* framework [33.46], in which a multitude of small HOG [33.47] detectors (Sect. 33.2.4) are combined in order to detect people.

Since, in theory, the ISM voting space can accommodate arbitrary quantities in addition to object center and scale, there have been extensions to predict the walking poses of pedestrians [33.48] as well as predicting the viewpoint from which the detected object is imaged [33.49, 50].

Pictorial Structures

While being the first to conceptualize the idea of objects being composed of flexible parts, the pictorial structures model [33.39, 51] is typically understood as a simplified version of the constellation model. The full mutual dependency structure between parts is relaxed to a tree (or a mixture of trees in more advanced incarnations) – note that the star-shaped ISM (Sect. 33.2.3) is hence a special case of a pictorial structures model with tree-depth one. In contrast to the constellation model, where parts are defined in a data-driven way, the pictorial structures model is typically associated with a semantic notion of parts. An intuitive example is the set of human body parts (head, torso, upper arm, lower arm, etc.) in connection with the natural kinematic tree structure (Fig. 33.3).

Avidbots Ex1016
Page 864 of 2154

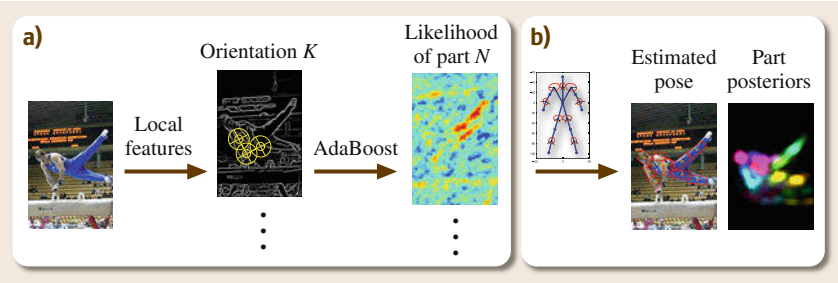


Fig.33.3a,b Human body pose estimation with the pictorial structures model. **(a)** Image likelihoods for individual parts; **(b)** final pose estimate using spatial layout prior

In fact, current state-of-the-art systems for people detection and human body pose estimation are typically built upon the pictorial structures model [33.52]. Here, pairwise relations between body parts express the angular constraints of the human body as well as the prior probability of observing people in various poses (e.g., it is unlikely to observe extreme angles at certain joints, Fig. 33.3b). In order to compensate for the weaker spatial model in comparison to the constellation model, the pictorial structures model typically relies on a strong model for the appearance of each part that is trained discriminatively (distinguishing the part from background, Fig. 33.3a) rather than in a generate way (explaining the part's appearance). Examples of discrimative part appearance models include AdaBoost [33.23] classifiers trained on dense shape context features [33.52]. Also note that part appearance and spatial layout models are trained independently (joint training has been realized to bear large potential for improvement in the context of the deformable part model [33.53] which we will review in Sect. 33.2.5).

The use of semantic parts naturally demands to provide additional supervision in the form of part annotations during training; human body parts are typically annotated with a fixed size relative to the size of the person. For recognition, inference in the tree-structured probabilistic graphical model can be carried out exactly and efficiently by max-product or sum-product belief propagation, depending on whether the maximum a posteriori (MAP) or marginal solution is preferred.

Extensions to the pictorial structures model include learning the tree structure instead of using the natural kinematic tree [33.54], introducing additional lateral edges in addition to the tree edges [33.55], and combining pictorial structures models with nonparametric representations of partial human poses (*poselets* [33.46]).

33.2.4 Histogram of Oriented Gradients

At the opposite end of the spectrum from the bag-of-words (Sect. 33.2.1) or flexible part models (Sect. 33.2.3) are approaches that represent an object class as a rigid template of local features with a fixed spatial layout. Conceptually, this corresponds to the finest level of a spatial pyramid in which each feature gets assigned its own exclusive set of histogram bins (with values equal to the feature itself). The by far most successful rigid template detector is the histogram of oriented gradients (HOG) [33.47] detector, and we will discuss its properties in detail in this section. Originally developed for people detection, the HOG feature has evolved into one of the most frequently used features in computer vision to date, and also constitutes the basis for the successful deformable part model (DPM) [33.53] described in Sect. 33.2.5.

Feature Extraction

The HOG feature can be seen as an extension of SIFT [33.9] from a single circular interest region to a rectangular detection window – at its core, it bins gradient directions into histograms, weighting their contributions by their respective magnitudes. Like SIFT, it has been carefully engineered and its parameters empirically tuned for performance (more recently, learning features end-to-end has received renewed attention, Sect. 33.2.6).

The first step for the computation of a HOG feature is to divide the detection window into a fixed grid of cells: a cell is the basic spatial unit of measurement for the HOG feature. The second step is to gamma-normalize the pixel content of each cell, in order to increase robustness to lighting changes. Third, the image is smoothed and image gradients computed using discrete derivative masks. Fourth, gradients are binned, such that each gradient casts weighted votes (by magnitude) according to nine different gradient orientation bins. Votes are interpolated bilinearly in both orientation and position. And fifth, the per-cell histograms are contrast normalized with respect to a larger block of surrounding cells in order to compensate for local variations in lighting and foreground–background contrast. Crucially, blocks are formed such that they overlap, which means that each cell contributes multiple times to the final feature descriptor, normalized in different ways, resulting in an over-complete represen-

Avidbots Ex1016
Page 865 of 2154

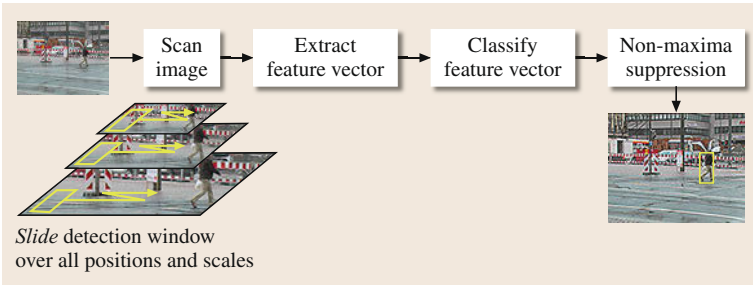


Fig. 33.4 Sliding window object class detection

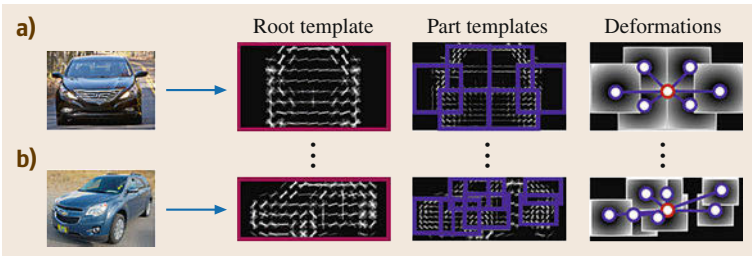


Fig.33.5a,b The deformable part model with multiple mixture components. **(a)** Mixture component for a frontal view of object class car. **(b)** Mixture component for a side view of object class car (after [33.56])

tation. This step has been found to be essential for good performance.

Sliding Window Object Detection

Similar to pure bag-of-words and spatial pyramid-based detectors, the HOG detector is typically run in a sliding window fashion over a test image, over different locations and scales (Fig. 33.4). That is, HOG features are computed for a detection window of fixed size, resulting in a feature vector that can then be fed into a discriminative classifier, such as a linear support vector machine (SVM).

Bootstrapping

One of the key ingredients for good detection performance of the HOG detector [33.47] is a technique termed as *bootstrapping*(not to be confused with a technique of the same name known from statistics, in which a dataset is repeatedly subsampled). An initially trained model is refined by re-training it on an extended set of negative training images, similar in spirit to boosting [33.23]. Specifically, the initial model is run on a validation set, and the false positive detections (wrongly detected objects) are added to the negative training set. As a result, the decision boundary of the retrained classifier is pushed further away from these *hard negatives*. Note that this procedure is not guaranteed to always lead to an improvement, but often improves performance substantially in practice.

Extensions

Due to its simple grid structure, the HOG feature lends itself to being combined with other feature channels extracted from a similar spatial grid. Notably, it has been successfully combined with optical flow features for people detection in video [33.57].

33.2.5 The Deformable Part Model

The section provides an introduction to one of the most successful computer vision algorithms to date, the DPM [33.53]. Its success is partly due to its consistently high performance for detecting a wide variety of object classes, but also due to the fact that a high quality implementation has been made publicly available from the start. The original implementation has constantly been improved and updated, and even found its way into OpenCV [33.58]. The HOG [33.47] implementation at its core has found wide spread use throughout the computer vision community.

Model Structure and Inference

The DPM can be seen as a proper generalization of the fully rigid HOG [33.47] template to a flexible, part-based object class representation (Fig. 33.5). It generalizes HOG in two different ways that both increase its ability to handle intraclass variation: first, it divides the single HOG template into a collection of multiple templates that can move relative to one another and capture small variations in object shape (hence the name *Deformable Part Model*). Specifically, it includes a low-resolution *root* template for the entire object plus a number of high-resolution *parts* that are constrained to occur not too far from their expected positions relative to the root – this constitutes a star-shaped probabilistic graphical model (PGM) in full analogy

to the implicit shape model [33.44] (Sect. 33.2.3) or a special case of a tree-structured pictorial structures model [33.39] (Sect. 33.2.3). While this is less powerful than a full constellation model [33.41] in terms of representation, it has proven very effective in connection with a discriminative learning regime.

Second, the DPM is formulated as a mixture model with multiple components, where each component takes care of another aspect of the object class of interest, such as a particular viewpoint or a particular context in which the object occurs (e.g., a person sitting on a chair). Each component consists of a star-model with root and multiple part templates as described earlier, trained to respond the strongest to its dedicated aspect – this separation of concerns leads to a simpler learning problem than trying to handle all aspects with a single component.

For recognition, each of the DPM components produces a score as follows: first, all of the component's templates (root and parts) are convolved with the image in a sliding window fashion, as in the original HOG [33.47] work (Sect. 33.2.4), resulting in dense score maps over image locations and scales. Second, root and part scores are combined with spatial deformation terms that reflect the star-shaped spatial dependency structure, resulting in new score maps over locations and scales – note that this inference step can be carried out by max-product belief propagation, which is exact and efficient due to the star topology, and implemented as an instance of a generalized distance transform [33.59]. And third, for each image location and scale, the maximum response over all components is kept as the final detection score (which is then further processed by a standard, greedy nonmaxima suppression).

Learning

It has been argued that much of the DPM's excellent detection performance is due to the specific way in which parameters are learned from training data, in addition to the added expressive power in comparison to a single, rigid HOG template. This learning also constitutes the major difference between the DPM and earlier flexible part models such as the ISM [33.44] or the pictorial structures model [33.39] (Sect. 33.2.3) that are structurally equivalent, but less effective in terms of performance.

In contrast to these earlier models, where local part appearance and spatial layout models are trained separately, all DPM parameters are learned jointly, such that a single binary classification objective is optimized (distinguish between the object class of interest and background). Crucially, this includes the placement of parts relative to the root template as well as the selection of the mixture component that is deemed responsible; both are treated as latent variables at training (and test) time. As a consequence, all aspects of the DPM are optimized towards the single goal of object class detection, without being convoluted by potentially suboptimal choices of part placement based on appearance similarity (ISM) or semantic meaning (pictorial structures).

The joint optimization problem is phrased as a generalization of the linear SVM [33.21] formulation of a single HOG template, termed latent support vector machine (LSVM) [33.60]. It is based on the following intuition: assuming that the latent variables (part placements and mixture component selection) are observed, all model parameters (HOG template and part displacement weights) can be learned using standard linear SVM procedures (the resulting objective function is convex, assuming the use of a hinge loss). Conversely, assuming known model parameters, the optimal values of all hidden variables can be found exactly and efficiently (using the distance transform for part placement and exhaustive search over components). This intuition gives rise to an EM-like alternation of two steps, which can be shown to converge to a local optimum of the original, nonconvex objective function.

Naturally, a good initialization is needed in order to reach good local optima during learning. The DPM suggests a clustering of training examples based on aspect ratio, which has proven to work surprisingly well in practice (clustering based on appearance typically results in small improvements but is much more costly in terms of computation time). As for the HOG detector [33.47], bootstrapping (here called *hard negative mining*) allows to escape bad local optima.

While the concept of jointly optimizing part appearance and placement has already existed in the constellation model [33.41], the DPM achieves this goal by largely different technical means, resulting in better performance: (1) it uses sliding-window HOG templates with linear SVM classifiers instead of a generative model over local interest point descriptors to represent local appearance, and (2) it relaxes the fully connected probabilistic graphical model topology to a star, making exact inference tractable and efficient for learning. The DPM has achieved excellent performance on a wide variety of benchmark datasets, including PASCAL VOC (visual object class) [33.61] (Sect. 33.2.8).

Multiview and 3-D DPMs

While the original DPM [33.53] achieves remarkable performance on standard detection benchmarks, it is inherently limited to predicting 2-D bounding boxes. While this might be sufficient for some applications (such as counting the number of objects in an image), it

Avidbots Ex1016
Page 867 of 2154

is clearly not enough for others: suppose an autonomous driving scenario in which vehicles are to be tracked across multiple frames, and their spatial extent to be estimated in order to avoid collisions – here, a 3-D representation of each vehicle is required.

To that end, the DPM has been extended [33.56, 62, 63] to output richer object hypotheses than 2-D bounding boxes, in the form of additional viewpoint estimates (providing a better notion of 3-D object extent than a 2-D bounding box alone) and 3-D parts that are consistent across viewpoints (providing a basis for establishing correspondences across multiple frames in a sequence). Providing these richer outputs can be achieved by extending the original DPM in two different ways: first, the classification problem solved during learning can be reformulated as a structured output prediction problem [33.62]. In this way, the model can be optimized for the prediction of other quantities than a binary object-background label, such as 2-D bounding box overlap or angular viewpoint estimates, through the use of different loss functions. In its simplest variant, viewpoint estimation can be implemented by a binning of angles, and dedicating a single DPM mixture component to each angular bin. While this model is still inherently view-based, it has been shown to deliver excellent performance in both 2-D bounding box localization and viewpoint estimation (note that viewpoint labels have to be available during training in addition to bounding box annotations).

Second, instead of parameterizing the DPM parts in the image plane, an object-centric 3-D coordinate system can be used, such that parts from different mixture components refer to the same 3-D object portion (Fig. 33.6a). Part deformations are then governed by 3-D displacement distributions (Fig. 33.6b), and object as well as part appearance can be captured in a continuous appearance model (Fig. 33.7a). The result is a 3-D object class representation [33.63] that can synthesize previously unseen viewpoints at test time (by interpolating between a number of supporting viewpoints for which training data is available) and thus estimate viewpoints up to arbitrary granularity. The learning of this 3-D object class representation is facilitated by using 3-D computer-aided design (CAD) models of the object class of interest in addition to real-world images (Fig. 33.7b). During training, the CAD models serve both to establish a 3-D coordinate system and to generate artificial training data for different viewpoints through rendering. Figure 33.8 shows example detections obtained by the 3-D^{2}PM.

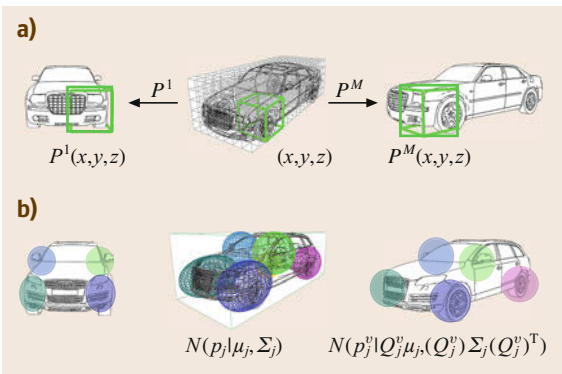


Fig.33.6a,b Part representation in the 3-D deformable part model (3-D DPM). **(a)** 3-D/2-D part coordinate systems; (after [33.62]) **(b)** 3-D/2-D part spatial distributions (after [33.63])

Computational Considerations

Interestingly, it has been realized that the major computational bottleneck of the original DPM [33.53] at recognition time is the convolution of HOG templates with image features (and not probabilistic graphical model inference as one might think). As a consequence, there have been attempts to minimize the number of HOG convolutions by various means, e.g., by part sharing parts across components and object classes [33.64], re-using parts of templates by decomposing them into an overcomplete set of dictionary elements [33.65] or cascaded detector evaluation [33.66]. Likewise, there have been attempts to reduce the training time for the DPM by exploiting HOG feature statistics to re-use computation across models [33.67, 68].

33.2.6 Convolutional Neural Networks

While artificial neural networks (ANNs) [33.69] had largely been neglected in favor of support vector machines, random forests, and other learners until around 2010, they have recently returned to the scene, with quite remarkable success. Among a number of artificial

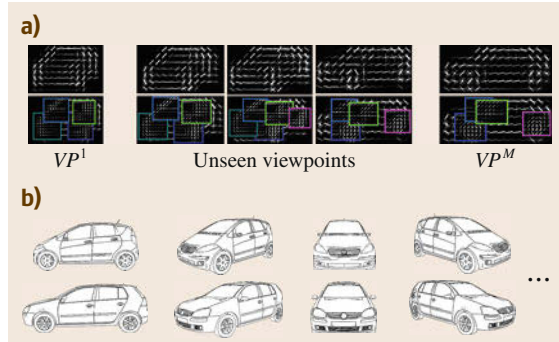


Fig.33.7a,b Continuous appearance model **(a)** in the 3-D^{2}PM, trained from rendered 3-D CAD models (after [33.63]) **(b)** and real images (not shown)

Avidbots Ex1016
Page 868 of 2154

neural network architectures that had been suggested already in the early days of computer vision, the convolutional neural network (CNN) [33.70] has emerged as a particularly promising candidate for becoming a standard tool for visual recognition tasks.

One of the key properties of the CNN is its specific structure (Fig. 33.9), which is both amenable to dealing with multidimensional input (such as a color image) and effective for preventing overfitting when learned from limited training data. As an ANN, a CNN defines an artificial neural network with multiple layers on an input image, such that image pixels provide the input to the neurons of the first layer, and the outputs of each layer of neurons provide the inputs to the next layer (the final layer can provide a task specific output, such as an encoding of different image classes). Now, the topology of the CNN is restricted in two specific ways: first, neurons in one layer can only be connected to a fixed spatial region of neurons in the preceding layer (the *receptive field*) – this dramatically reduces the total number of connections in the network and hence parameters of the model that have to be learned. And second, most layers of neurons (typically the lower layers) are formed by replicating local groups of neurons multiple times, including the weights associated with their connections to the preceding layer (*weight sharing*) – this further reduces the number of parameters of the model, and encourages the network to learn a representation for the input that is invariant to translation. As a result of their specific structure, CNNs are less prone to overfitting than fully connected ANNs, and can be successfully trained using gradient-based methods and back-propagation of errors.

CNNs for Image Classification

CNNs have recently achieved remarkable performance in large-scale image classification [33.71], fueled by the availability of large-scale datasets like ImageNet [33.33] that provide thousands of images per object class. The success of CNNs in image classification is mostly attributed to their ability to learn useful image representations end-to-end (*deep learning*) rather than relying on engineered features, such as SIFT [33.9] or HOG [33.47]. While the latter have been carefully designed and empirically tuned over many years, they are *general purpose* and not specifically adapted to the classification task at hand, and hence suboptimal. A recent study provides further insights into the operation of individual CNN components (hidden layers and classifiers) through visualization [33.74], resulting in a further boost in performance.

Region Proposals for Detection

Since the evaluation of a CNN is computationally expensive due to convolutions, a CNN is typically applied to preselected candidate regions rather than in a dense sliding window fashion for detection. Candidate regions can be generated either by a specialized detector trained for the object class of interest (such as HOG [33.47], Sect. 33.2.4) that is weaker but also much faster to evaluate than the CNN, or by a more general attention mechanism that generates bounding boxes that are likely to contain an object instance of *any* class. Among various methods that generate proposal regions with a high degree of *objectness* [33.75] ([33.76] for a recent discussion and analysis of region proposal methods), selective search [33.77] has been successfully used in



Fig. 33.8 3-D^{2}PM example detections (car, bicycle, human head, cell phone)

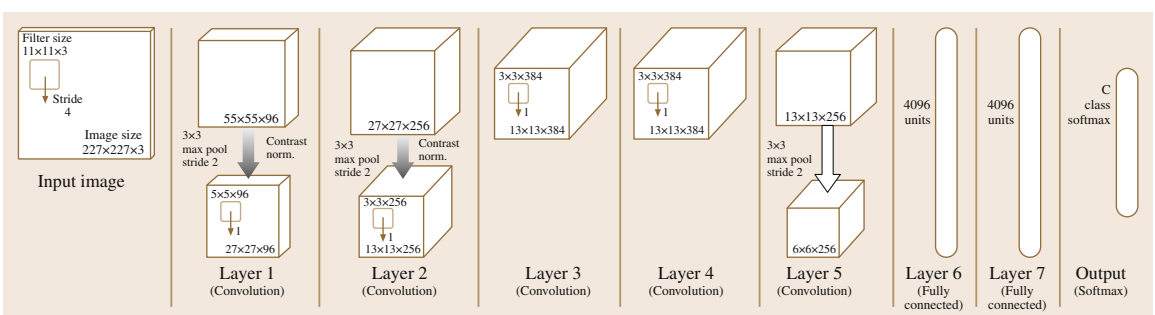


Fig. 33.9 The CNN model of (after [33.71]) in the implementation of Caffe (after [33.72, 73])

Avidbots Ex1016
Page 869 of 2154

connection with CNN-based, object class specific classifiers for detection [33.78]. Selective search is built upon a superpixel-based oversegmentation of an image, and suggests a greedy procedure to group adjacent superpixels into regions of high objectness. The resulting detector [33.79] has been shown to outperform previous results in generic object class detection by a large margin (notably outperforming the previous best results of the DPM [33.53], Sect. 33.2.5, on PASCAL VOC [33.61], Sect. 33.2.8). Curiously, it is not necessary to train this detector end-to-end for the task of object class detection for good performance – instead, the lower layers of the CNN can be pretrained for classification on ImageNet as in [33.71], and later fine-tuned for detection. Both the CNN pretrained on ImageNet and the resulting state-of-the-art object class detector are publicly available under the name *Caffe* [33.72] and likely to find their way into the toolboxes of many computer vision practitioners.

Structured Outputs

While artificial neural networks can represent any function in theory, effective learning of ANNs with more complex, structured outputs than single class labels or continuous values remains a challenge. Instead of learning a full network end-to-end, structured outputs are typically obtained by feeding the outputs of the last layer of neurons into an existing structured model, such as multiple body joint regressors for human body pose estimation [33.80].

33.2.7 Fine-Grained Categorization

In recent years, there has been a growing interest in distinguishing objects not only with respect to basic-level categories [33.5] (such as car, bicycle, and person), but also on a finer level of granularity (such as different car models or plant and animal species). This *fine-grained categorization* is deemed particularly challenging: the distinction between two categories is often determined by small visual differences (such as the shape of a car's headlight): these differences are easily outweighed by appearance differences due to change in lighting or viewpoint (a car looks largely different from the front and from the side), bearing the risk of overfitting to irrelevant details.

Feature Selection

A first strategy to identifying subtle discriminative features while at the same time preventing overfitting is to combine classical feature selection techniques with randomization, e.g., in the form of random forest classifiers [33.35, 81]. Here, randomization acts as a regularizer that prevents multiple trees to latch onto the same potentially irrelevant feature, hence reducing the risk of overfitting. Feature selection can also be performed with humans in the loop, using crowd-sourcing and gamification [33.82].

Pose Normalization

A second, orthogonal strategy in fin-grained categorization is to devise factored object class representations that explicitly capture different modes of variation in object appearance, thereby separating variations caused by category membership from variations caused by other factors, such as viewpoint changes. The main idea is to establish a common frame of reference for the extraction of visual features (e.g., a local coordinate system for the dense extraction of SIFT [33.9] features), such that the same object instance will always result in (roughly) the same visual features being extracted, irrespective of viewpoint, lighting, etc.

Specifically, such a frame of reference can be established by first detecting the object of interest in a test image and also localizing its semantic parts. Visual features are then extracted and described relative to the estimated object position and detected parts, ensuring that a feature on the beak of a bird will only ever be compared to a corresponding feature on the beak of another bird, and not on its tail (assuming perfect detection, viewpoint estimation, and part localization). This kind of part-based *pose normalization* has been demonstrated to be effective for bird species categorization [33.83, 84].

Instead of localizing object parts in 2-D, some works establish a common reference frame by estimating a rough 3-D geometry of the object of interest prior to feature extraction, in addition to estimating its viewpoint. This allows not only to describe feature locations relative to the object surface rather than the 2-D image plane (effectively performing the pooling in 3-D [33.36]), but also to rectify image patches with respect to the estimated surface normals, improving the robustness of the resulting feature descriptors with respect to small viewpoint changes. This strategy has proven effective in connection with a prebuilt bird model [33.35] as well as in connection with predicted 3-D CAD model matches for fine-grained car categorization [33.36].

33.2.8 Datasets

Since visual object class recognition approaches rely on machine learning algorithms at their core, they are very much dependent on the availability of sufficient training data – it has in fact been argued that major advances

Avidbots Ex1016
Page 870 of 2154

in the field might have occurred in response of obtaining more data rather than by developing more powerful algorithms [33.85]. At the same time, datasets are of utmost importance in order to quantify and compare the performance of different methods under controlled conditions, and checking the sanity of newly developed algorithms. To that end, benchmark datasets for visual object class recognition typically consist of a collection of images split into training, validation, and test sets, plus a set of ground truth annotations, and often also evaluation scripts that measure the performance of a recognition algorithm relative to the provided annotations.

Since each dataset provides only a small sample of the visual world, there is an inherent danger of being biased [33.85] with respect to which images are selected to be part of a dataset (and which ones are not), and by what means images are captured. As a result, new recognition algorithms should always be tested and cross-trained on multiple datasets in order to prevent overfitting to the characteristics of a particular dataset rather than learning a concept of interest. The following paragraphs give a small, by no means complete, selection of some of today's influential datasets for visual object class recognition.

PASCAL VOC

The PASCAL visual object classes challenge [33.61] stands for a succession of datasets that have been released on a yearly basis from 2005 to 2012. Each dataset consists of multiple challenges (image classification, object class detection, segmentation, human body pose estimation) that are defined on a given set of images. Notably, the set of images is steadily increased, such that images of year n are re-used in year $n+1$. PASCAL VOC counteracts dataset overfitting by not releasing test set ground truth annotations to the public, but instead maintaining an evaluation server that runs a limited number of automated tests for a particular recognition algorithm. PASCAL VOC images have been equipped with additional annotations, e.g., in the form of aligned 3-D CAD models (3-D-PASCAL [33.86]).

SUN and COCO

The scene understanding (SUN) [33.87] and common objects in context (COCO) [33.88] datasets have been built in an effort to capture entire visual scenes rather than objects in isolation. They are meant as benchmarks for methods that reason jointly about multiple objects in a scene rather than performing independent classification or detection.

ImageNet

ImageNet [33.33] is one of the largest image datasets to date, consisting of over 14 million internet images that have been annotated with object labels from the WordNet [33.89] hierarchy. Like PASCAL VOC, it constitutes the basis for multiple challenges that are defined on the dataset, ranging from classification (1000 classes) over object class detection to fine-grained categorization. Due to its massive size, the collection of the ImageNet dataset has been performed mostly by means of crowd sourcing. ImageNet constitutes the basis for pretraining some of the most successful CNN-based image classifiers to date [33.71] (Sect. 33.2.6).

33.3 Discussion and Conclusions

In this chapter, we have discussed visual object class recognition as one of the most fundamental problems in computer vision, with a focus on presenting the most promising models available to practitioners today. The selection of methods that we have reviewed is by no means complete – in favor of going into some depth for the most common techniques, we have skipped over object representations based on manifold learning [33.90] as well as representations based on grammars [33.91, 92], to mention just a few.

As we have seen, practically all available models are inherently based on some notion of object parts that are governed by certain topological constraints. Naturally, the choice of which part-based model to pick is not an easy one, as it depends on many different factors that should be carefully considered.

33.3.1 Task

First of all, the appropriate choice of model is dependent on the application scenario and the resulting desired recognition output: different methods can generate outputs of different granularity, ranging from image-level class labels (basic bag-of-words, Sect. 33.2.1, spatial pyramid matching, Sect. 33.2.2) over 2-D bounding box locations (bag-of-words and spatial pyramid matching run in sliding window fashion, histogram of oriented gradient detectors, Sect. 33.2.4, flexible part models, Sect. 33.2.3, basic deformable part models, Sect. 33.2.5) to multiview detections and viewpoint estimates (multiview ISM, Sect. 33.2.3, 3-D DPM, Sect. 33.2.5).

Secondly, an application scenario typically comes with a certain set of object classes of interest, which

Avidbots Ex1016
Page 871 of 2154

are in turn characterized by rather different visual properties. Rigid objects (like cars or mugs) tend to be handled well by rigid templates (like histogram of oriented gradients, Sect. 33.2.4) or deformable part models (Sect. 33.2.5), while articulate objects (like cats and dogs) can be more reliably represented using looser spatial models like bag of words (Sect. 33.2.1), or by explicitly modeling their kinematic tree structure (pictorial structures, Sect. 33.2.3).

Finally, it should be noted that the distinction between object classes itself is quite arbitrary when drawing strict borders between any two classes. In reality, some classes are inherently more similar than others (e.g., dogs and horses are more similar than dogs and cars, since dogs and horses are both quadrupeds). Looking at multiple levels of object categorization rather than individual classes, it becomes a desired property that objects from the same super-ordinate category, such as quadrupeds, be classified as more similar than objects from different super-ordinate categories. In other words, an object categorization system should degrade gracefully: if the object itself is not correctly recognized, then we want it to be assigned at least to a *similar* category. We thus need a way of representing the relationship and grading similarities between object categories [33.93].

33.3.2 Data

A particular choice of model implies a certain amount of parameters that have to be learned and additional latent parameters that have to be inferred at test time. A model has to be chosen not only appropriate for the task (above) but also in accordance with the available data at hand in order to prevent over or under fitting.

33.3.3 Efficiency

While simple *feedforward* models like histograms of oriented gradients (Sect. 33.2.4) and bag-of-words (Sect. 33.2.1) are very efficient, modeling deformations and interdependence of parts (flexible part models, Sect. 33.2.3, deformable part models, Sect. 33.2.5) comes at the cost of inferring latent quantities in the model. Notably, this inference often has to be performed repeatedly during learning, underlining the importance of efficient inference procedures.

References

33.1 D. Marr: *Vision* (Freeman, San Francisco 1982)
33.2 D.H. Ballard, C.M. Brown: *Computer Vision* (Prentice Hall, Englewood Cliffs 1982)
33.3 R. Brown: How shall a thing be called?, Psychol. Rev. **65**, 14–21 (1958)
33.4 R. Brown: *Social Psychology* (Free, New York 1965)
33.5 E. Rosch, C. Mervis, W. Gray, D. Johnson, P. Boyes-Braem: Basic objects in natural categories, Cogn. Psychol. **8**, 382–439 (1976)
33.6 G. Lakoff: *Women, Fire, and Dangerous Things – What Categories Reveal About the Mind* (Univ. Chicago Press, Chicago 1987)
33.7 S. Dickinson, A. Leonardis, B. Schiele, M. Tarr: *Object Categorization: Computer and Human Vision Perspectives* (Cambridge Univ. Press, Cambridge 2009)
33.8 G. Csurka, C.R. Dance, L. Fan, J. Willarnowski, C. Bray: Visual categorization with bags of keypoints, Eur. Conf. Comput. Vis. (ECCV) (2004)
33.9 D. Lowe: Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. **60**(2), 91–110 (2004)
33.10 K. Mikolajczyk, C. Schmid: A performance evaluation of local descriptors, IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2004)
33.11 J. Matas, O. Chum, M. Urban, T. Pajdla: Robust wide baseline stereo from maximally stable extremal regions, Image Vis. Comput. **22**(10), 761–767 (2004)
33.12 T. Tuytelaars: Dense interest points, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2010)
33.13 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification* (Wiley, New York 2000)
33.14 F. Jurie, B. Triggs: Creating efficient codebooks for visual recognition, 10th IEEE Int. Conf. Comput. Vis. (ICCV) (2005)
33.15 B. Schiele, J.L. Crowley: Recognition without correspondence using multidimensional receptive field histograms, Int. J. Comput. Vis. **36**(1), 31–52 (2000)
33.16 T. Hofmann: Probabilistic latent semantic indexing, Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. (1999)
33.17 D.M. Blei, A.Y. Ng, M.I. Jordan: Latent dirichlet allocation, J. Mach. Learn. Res. **3**, 983–1022 (2003)
33.18 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. R. Statist. Soc. B **39**, 1–38 (1977)
33.19 Z. Shi, Y. Yang, T.M. Hospedales, T. Xiang: Weakly supervised learning of objects, attributes and their associations, Eur. Conf. Comput. Vis. (ECCV) (2014) pp. 472–487
33.20 S. Lazebnik, C. Schmid, J. Ponce: Spatial pyramid matching. In: *Object Categorization*, ed. by S. Dickinson, A. Leonardis, B. Schiele, M. Tarr (Cambridge Univ. Press, Cambridge 2009) pp. 401–415
33.21 C. Cortes, V. Vapnik: Support-vector networks, Mach. Learn. **20**(3), 273–297 (1995)
33.22 S. Maji, A.C. Berg, J. Malik: Classification using intersection kernel support vector machines is efficient, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2008)
33.23 Y. Freund, R.E. Schapire: A decision-theoretic generalization of on-line learning and an application

Avidbots Ex1016
Page 872 of 2154

to boosting, J. Comput. Syst. Sci. **55**(1), 119–139 (1997)
33.24 L. Breiman: Random forests, Mach. Learn. **45**(1), 5–32 (2001)
33.25 T. Lindeberg: Feature detection with automatic scale selection, Int. J. Comput. Vis. **30**(2), 79–116 (1998)
33.26 C.H. Lampert, M.B. Blaschko, T. Hofmann: Efficient subwindow search: A branch and bound framework for object localization, IEEE Trans. Pattern Anal. Mach. Intell. **31**(12), 2129–2142 (2009)
33.27 S. Lazebnik, C. Schmid, J. Ponce: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2006)
33.28 V. Ferrari, L. Fevrier, F. Jurie, C. Schmid: Groups of adjacent contour segments for object detection, IEEE Trans. Pattern Anal. Mach. Intell. **30**(1), 36–51 (2008)
33.29 E. Shechtman, M. Irani: Matching local self-similarities across images and videos, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2007)
33.30 F. Perronnin, C. Dance: Fisher kernels on visual vocabularies for image categorization, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2007)
33.31 J. Yang, K. Yu, Y. Gong, T. Huang: Linear spatial pyramid matching using sparse coding for image classification, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2009)
33.32 J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong: Locality-constrained linear coding for image classification, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2010)
33.33 J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei: ImageNet: A large-scale hierarchical image database, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2009)
33.34 N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A.J. Rodriguez-Sanchez, L. Wiskott: Deep hierarchies in the primate visual cortex: What can we learn for computer vision?, IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1847–1871 (2013)
33.35 R. Farrell, O. Oza, N. Zhang, V.I. Morariu, T. Darrell, L.S. Davis: Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance, IEEE Int. Conf. Comput. Vis. (ICCV) (2011)
33.36 J. Krause, M. Stark, J. Deng, L. Fei-Fei: 3d object representations for fine-grained categorization, 4th Int. IEEE Workshop 3D Represent. Recognit. (3dRR-13), Sydney (2013)
33.37 M. Varma, D. Ray: Learning the discriminative power-invariance trade-off, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2007)
33.38 A. Vedaldi, V. Gulshan, M. Varma, A. Zisserman: Multiple kernels for object detection, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.39 M.A. Fischler, R.A. Elschlager: The representation and matching of pictorial structures, IEEE Trans. Comput. **22**(1), 67–92 (1973)
33.40 M. Weber, M. Welling, P. Perona: Unsupervised learning of models for recognition, Eur. Conf. Comput. Vis. (ECCV) (2000)
33.41 R. Fergus, P. Perona, A. Zisserman: Object class recognition by unsupervised scale-invariant learning, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2003)
33.42 M. Stark, M. Goesele, B. Schiele: A shape-based object class model for knowledge transfer, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.43 L. Fei-Fei, R. Fergus, P. Perona: Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2004) pp. 178–186
33.44 B. Leibe, A. Leonardis, B. Schiele: Robust object detection by interleaving categorization and segmentation, Int. J. Comput. Vis. **77**(1–3), 259–289 (2008)
33.45 S. Maji, J. Malik: Object detection using a max-margin Hough transform, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2009)
33.46 L. Bourdev, J. Malik: Poselets: Body part detectors trained using 3D human pose annotations, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.47 N. Dalal, B. Triggs: Histograms of oriented gradients for human detection, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2005) pp. 886–893
33.48 B. Leibe, E. Seemann, B. Schiele: Pedestrian detection in crowded scenes, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Washington (2005) pp. 878–885
33.49 A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, L. Van Gool: Towards multi-view object class detection, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2006)
33.50 M. Arie-Nachimson, R. Basri: Constructing implicit 3D shape models for pose estimation, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.51 P.F. Felzenszwalb, D.P. Huttenlocher: Efficient matching of pictorial structures, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2000)
33.52 M. Andriluka, S. Roth, B. Schiele: Pictorial structures revisited: People detection and articulated pose estimation, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2009)
33.53 P.F. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan: Object detection with discriminatively trained part based models, IEEE Trans. Pattern Anal. Machin. Intell. **32**(9), 1627–1645 (2010)
33.54 F. Wang, Y. Li: Beyond physical connections: Tree models in human pose estimation, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2013) pp. 596–603
33.55 M. Sun, M. Telaprolu, H. Lee, S. Savarese: An efficient branch-and-bound algorithm for optimal human pose estimation, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2012)
33.56 P. Bojan, M. Stark: Multi-view and 3D deformable part models, IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **37**(11), 2232–2245 (2015)
33.57 N. Dalal, B. Triggs, C. Schmid: Human detection using oriented histograms of flow and appearance, Eur. Conf. Comput. Vis. (ECCV) (2006)
33.58 G. Bradski: Opencv. http://opencv.org/ (July 09, 2015)

Avidbots Ex1016
Page 873 of 2154

33.59 P.F. Felzenszwalb, D.P. Huttenlocher: *Distance transforms of sampled functions, Technical Report 1963* (Cornell Univ., Ithaca 2004)

33.60 C.-N.J. Yu, T. Joachims: Learning structural SVMs with latent variables, ACM Proc. 26th Annu. Int. Conf. Mach. Learn., New York (2009) pp. 1169–1176

33.61 M. Everingham, L. Gool, C.K. Williams, J. Winn, A. Zisserman: The PASCAL visual object classes (VOC) challenge, Int. J. Comput. Vis. **88**(2), 303–338 (2010)

33.62 B. Pepik, M. Stark, P. Gehler, B. Schiele: Teaching 3d geometry to deformable part models, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2012)

33.63 B. Pepik, P. Gehler, M. Stark, B. Schiele: $3D^2PM$–3D deformable part models, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.64 P. Ott, M. Everingham: Shared parts for deformable part-based models, IEEE Comput. Vis. Pattern Recognit. (CVPR) (2011)

33.65 H.O. Song, S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, T. Darrell: Sparselet models for efficient multiclass object detection, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.66 P. Felzenszwalb, R. Girshick, D. McAllester: Cascade object detection with deformable part models, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2010)

33.67 T. Gao, M. Stark, D. Koller: What makes a good detector? – structured priors for learning from few examples, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.68 B. Hariharan, J. Malik, D. Ramanan: Discriminative decorrelation for clustering and classification, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.69 S. Haykin: *Neural Networks: A Comprehensive Foundation*, 2nd edn. (Prentice Hall, Upper Saddle River 1998)

33.70 Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel: Backpropagation applied to handwritten zip code recognition, Neural Comput. **1**(4), 541–551 (1989)

33.71 A. Krizhevsky, I. Sutskever, G.E. Hinton: Imagenet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. **25**, 1097–1105 (2012)

33.72 Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell: Caffe: Convolutional architecture for fast feature embedding, http://caffe.berkeleyvision.org/ (arXiv preprint arXiv:1408.5093) (2014)

33.73 J. Hosang, M. Omran, R. Benenson, B. Schiele: Taking a deeper look at pedestrians, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2015)

33.74 M.D. Zeiler, R. Fergus: Visualizing and understanding convolutional networks, Eur. Conf. Comput Vis. (ECCV) (2014)

33.75 B. Alexe, T. Deselares, V. Ferrari: Measuring the objectness of image windows, IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2189–2202 (2012)

33.76 J. Hosang, R. Benenson, B. Schiele: How good are detection proposals, really?, 25th Br. Mach. Vis. Conf. (BMVC) (2014)

33.77 K.E.A. van de Sande, J.R.R. Uijlings, T. Gevers, A.W.M. Smeulders: Segmentation as selective search for object recognition, IEEE Int. Conf. Comput. Vis. (ICCV) (2013)

33.78 B. Hariharan, P. Arbeláez, R. Girshick, J. Malik: Simultaneous detection and segmentation, Eur. Conf. Comput. Vis. (ECCV) (2014)

33.79 R. Girshick, J. Donahue, T. Darrell, J. Malik: Rich feature hierarchies for accurate object detection and semantic segmentation, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2014)

33.80 A. Toshev, C. Szegedy: Deeppose: Human pose estimation via deep neural networks, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2014)

33.81 J. Deng, J. Krause, M. Stark, L. Fei-Fei: Leveraging the wisdom of the crowd for fine-grained recognition, IEEE Trans. Pattern Anal. Mach. Intel. (2015)

33.82 J. Deng, J. Krause, L. Fei-Fei: Fine-grained crowdsourcing for fine-grained recognition, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2013)

33.83 N. Zhang, R. Farrell, T. Darrell: Pose pooling kernels for sub-category recognition, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2012)

33.84 T. Berg, P.N. Belhumeur: Poof: Part-based one-vs-one features for fine-grained categorization, face verification, and attribute estimation, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2013)

33.85 A. Torralba, A.A. Efros: Unbiased look at dataset bias, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2011)

33.86 Y. Xiang, R. Mottaghi, S. Savarese: Beyond pascal: A benchmark for 3d object detection in the wild, IEEE Winter Conf. Appl. Comput. Vis. (WACV) (2014)

33.87 J. Xiao, J. Hays, K. Ehinger, A. Oliva, A. Torralba: Sun database: Large-scale scene recognition from abbey to zoo, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2010)

33.88 T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick: Microsoft coco: Common objects in context, Lect. Notes Comput. Sci. **8693**, 740–755 (2014)

33.89 G.A. Miller: Wordnet: A lexical database for english, ACM Communication **38**(11), 39–41 (1995)

33.90 M.A. Turk, A.P. Pentland: Face recognition using eigenfaces, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (1991) pp. 586–591

33.91 Z.W. Tu, X.R. Chen, A.L. Yuille, S.C. Zhu: Image parsing: Unifying segmentation, detection and recognition, Int. J. Comput. Vis. **63**(2), 113–140 (2005)

33.92 L. Zhu, Y. Chen, A.L. Yuille: Unsupervised learning of probabilistic Grammar–Markov models for object categories, IEEE Trans. Pattern Anal. Machin. Intell. **31**(1), 114–128 (2009)

33.93 J. Deng, J. Krause, A. Berg, L. Fei-Fei: Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Providence (2012)

Avidbots Ex1016
Page 874 of 2154

Multimedia Contents

34. Visual Servoing

François Chaumette, Seth Hutchinson, Peter Corke

This chapter introduces visual servo control, using computer vision data in the servo loop to control the motion of a robot. We first describe the basic techniques that are by now well established in the field. We give a general overview of the formulation of the visual servo control problem, and describe the two archetypal visual servo control schemes: image-based and pose-based visual servo control. We then discuss performance and stability issues that pertain to these two schemes, motivating advanced techniques. Of the many advanced techniques that have been developed, we discuss two-and-a-half-dimensional (2.5-D), hybrid, partitioned, and switched approaches. Having covered a variety of control schemes, we deal with target tracking and controlling motion directly in the joint space and extensions to underactuated ground and aerial robots. We conclude by describing applications of visual servoing in robotics.

34.1 **The Basic Components of Visual Servoing** 842
34.2 **Image-Based Visual Servo** 843
34.2.1 The Interaction Matrix 843
34.2.2 Approximating the Interaction Matrix 844
34.2.3 A Geometrical Interpretation of IBVS 846
34.2.4 Stability Analysis 847
34.2.5 IBVS with a Stereo Vision System 848
34.2.6 IBVS with Cylindrical Coordinates of Image Points 849
34.2.7 IBVS with Other Geometrical Features 849
34.2.8 Non-Perspective Cameras 850
34.2.9 Direct Estimation 850
34.3 **Pose-Based Visual Servo** 851
34.4 **Advanced Approaches** 854
34.4.1 Hybrid VS 854
34.4.2 Partitioned VS 855
34.5 **Performance Optimization and Planning** 856
34.5.1 Optimal Control and Redundancy Framework 856
34.5.2 Switching Schemes 856
34.5.3 Feature Trajectory Planning 857
34.6 **Estimation of 3-D Parameters** 858
34.7 **Determining s^* and Matching Issues** 859
34.8 **Target Tracking** 859
34.9 **Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space** 860
34.10 **Under Actuated Robots** 861
34.11 **Applications** 863
34.12 **Conclusions** 863
Video-References 863
References 863

Visual servo (VS) *control* refers to the use of computer vision data to control the motion of a robot. The vision data may be acquired from a camera that is mounted directly on a robot manipulator or on a mobile robot, in which case motion of the robot induces camera motion, or the camera can be fixed in the workspace so that it can observe the robot motion from a stationary configuration. Other configurations can be considered, such as

Avidbots Ex1016
Page 875 of 2154

for instance several cameras mounted on pan–tilt heads observing the robot motion. The mathematical development of all these cases is similar, and in this chapter we will focus primarily on the former, so-called *eye-in-hand*, case.

Visual servo control relies on techniques from image processing, computer vision, and control theory. In the present chapter, we will deal primarily with the issues from control theory, making connections to previous chapters when appropriate.

34.1 The Basic Components of Visual Servoing

The aim of all vision-based control schemes is to minimize an error $\boldsymbol{e}(t)$, which is typically defined by

$$\boldsymbol{e}(t) = \boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a}) - \boldsymbol{s}^* \,. \tag{34.1}$$

This formulation is quite general, and it encompasses a wide variety approaches, as we will see below. The parameters in (34.1) are defined as follows. The vector $\boldsymbol{m}(t)$ is a set of image measurements (e.g., the image coordinates of interest points, or the parameters of a set of image lines or segments). These image measurements are used to compute a vector of k visual features, $\boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a})$, in which $\boldsymbol{a}$ is a set of parameters that represent potential additional knowledge about the system (e.g., true or approximate camera intrinsic parameters or a model of the object to be tracked). The vector $\boldsymbol{s}^*$ contains the desired values of the features. Note that the order of the desired and actual values in (34.1) is reversed with respect to the common convention for feedback control systems.

For now, we consider the case of a fixed goal pose and a motionless target, i. e., $\boldsymbol{s}^*$ is constant, and changes in $\boldsymbol{s}$ depend only on camera motion. Further, we consider here the case of controlling the motion of a camera with six degrees of freedom (e.g., a camera attached to the end effector of a six-degree-of-freedom arm). We will treat more general cases in later sections.

Visual servoing schemes mainly differ in the way that $\boldsymbol{s}$ is designed. In Sects. 34.2 and 34.3, we describe classical approaches, including image-based visual servo control (IBVS), in which $\boldsymbol{s}$ consists of a set of features that are immediately available in the image, and pose-based visual servo control (PBVS), in which $\boldsymbol{s}$ consists of a pose, which must be estimated from image measurements. Note that in the older visual servoing literature PBVS is named position-based, rather than pose-based, visual servoing [34.1, 2]. We also present in Sect. 34.4 several more-advanced methods.

Once $\boldsymbol{s}$ is selected, the design of the control scheme can be quite simple. Perhaps the most straightforward approach is to design a velocity controller. To do this, we require the relationship between the time variation of $\boldsymbol{s}$ and the camera velocity. Let the spatial velocity of the camera be denoted by $\boldsymbol{v}_c = (\boldsymbol{v}_c, \boldsymbol{\omega}_c)$ where $\boldsymbol{v}_c$ is the instantaneous linear velocity of the origin of the camera frame and $\boldsymbol{\omega}_c$ is the instantaneous angular velocity of the camera frame. The relationship between $\dot{\boldsymbol{s}}$ and $\boldsymbol{v}_c$ is given by

$$\dot{\boldsymbol{s}} = \mathbf{L}_{\boldsymbol{s}} \boldsymbol{v}_c \,, \tag{34.2}$$

in which $\mathbf{L}_{\boldsymbol{s}} \in \mathbb{R}^{k\times 6}$ is called the *interaction matrix* related to $\boldsymbol{s}$ [34.3]. The term *feature Jacobian* is also used somewhat interchangeably in the visual servo literature [34.2], but in the present chapter we will use this latter term to relate the time variation of the features to the robot's joint velocity (Sect. 34.8).

Using (34.1) and (34.2) we immediately obtain the relationship between camera velocity and the time variation of the error

$$\dot{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{e}} \boldsymbol{v}_c \,, \tag{34.3}$$

where $\mathbf{L}_{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{s}}$. Considering $\boldsymbol{v}_c$ as the input to the robot controller, and if we would like, for instance, to design for an exponential and decoupled decrease of the error (i. e., $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$) then using (34.3) we obtain as controller

$$\boldsymbol{v}_c = -\lambda \mathbf{L}_{\boldsymbol{e}}^{+} \boldsymbol{e} \,, \tag{34.4}$$

where $\mathbf{L}_{\boldsymbol{e}}^{+} \in \mathbb{R}^{6\times k}$ is the Moore–Penrose pseudo-inverse of $\mathbf{L}_{\boldsymbol{e}}$, that is, $\mathbf{L}_{\boldsymbol{e}}^{+} = (\mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}} \mathbf{L}_{\boldsymbol{e}})^{-1} \mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}}$ when $k \geq 6$ and $\mathbf{L}_{\boldsymbol{e}}$ is of full rank 6. When $k = 6$, if $\det \mathbf{L}_{\boldsymbol{e}} \neq 0$ it is possible to invert $\mathbf{L}_{\boldsymbol{e}}$, giving the control $\boldsymbol{v}_c = -\lambda \mathbf{L}_{\boldsymbol{e}}^{-1} \boldsymbol{e}$. When $k \leq 6$ and $\mathbf{L}_{\boldsymbol{e}}$ is of full rank k, $\mathbf{L}_{\boldsymbol{e}}^{+}$ is given by

$$\mathbf{L}_{\boldsymbol{e}}^{+} = \mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}} \left(\mathbf{L}_{\boldsymbol{e}} \mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}} \right)^{-1} \,.$$

When $\mathbf{L}_{\boldsymbol{e}}$ is not full rank, the numerical value of $\mathbf{L}_{\boldsymbol{e}}^{+}$ can be obtained from the singular value decomposition of $\mathbf{L}_{\boldsymbol{e}}$. In all cases, control scheme (34.4) allows $\|\dot{\boldsymbol{e}} - \lambda \mathbf{L}_{\boldsymbol{e}} \mathbf{L}_{\boldsymbol{e}}^{+} \boldsymbol{e}\|$ and $\|\boldsymbol{v}_c\|$ to be minimal. Note that the desired behavior $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$ is obtained only when $\mathbf{L}_{\boldsymbol{e}} \mathbf{L}_{\boldsymbol{e}}^{+} = \mathbf{I}_k$, where $\mathbf{I}_k$ is the $k \times k$ identity matrix, that is, only when $\mathbf{L}_{\boldsymbol{e}}$ if of full rank k, $k \leq 6$.

In real visual servo systems, it is impossible to know perfectly in practice either $\mathbf{L}_{\boldsymbol{e}}$ or $\mathbf{L}_{\boldsymbol{e}}^{+}$. So an approximation or an estimation of one of these two matrices must

Avidbots Ex1016
Page 876 of 2154

be realized. In the sequel, we denote both the pseudo-inverse of the approximation of the interaction matrix and the approximation of the pseudo-inverse of the interaction matrix by the symbol $\widehat{\mathbf{L}_e^+}$. Using this notation, the control law is in fact

$$\boldsymbol{v}_c = -\lambda \widehat{\mathbf{L}_e^+} \boldsymbol{e} = -\lambda \widehat{\mathbf{L}_s^+} (\boldsymbol{s} - \boldsymbol{s}^*) \,. \tag{34.5}$$

Closing the loop and assuming that the robot controller is able to realize perfectly $\boldsymbol{v}_c$, that is inserting (34.5) into (34.3), we obtain

$$\dot{\boldsymbol{e}} = -\lambda \mathbf{L}_e \widehat{\mathbf{L}_e^+} \boldsymbol{e} \,. \tag{34.6}$$

This equation characterizes the actual behavior of the closed-loop system, which is different from the desired one ($\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$) when ever

$$\mathbf{L}_e \widehat{\mathbf{L}_e^+} \neq \mathbf{I}_k \,.$$

It is also the basis of the stability analysis of the system using Lyapunov theory.

What we have presented above is the basic design implemented by most visual servo controllers. All that remains is to fill in the details. How should $\boldsymbol{s}$ be chosen? What then is the form of $\mathbf{L}_s$? How should we estimate $\widehat{\mathbf{L}_e^+}$? What are the performance characteristics of the resulting closed-loop system? These questions are addressed in the remainder of the chapter. We first describe the two basic approaches, IBVS and PBVS, whose principles were proposed more than 20 years ago [34.1]. We then present more-recent approaches that have improved their performance.

34.2 Image-Based Visual Servo

Traditional image-based control schemes [34.1, 4] use the image-plane normalised coordinates of a set of points to define the vector $\boldsymbol{s}$. The image measurements $\boldsymbol{m}$ are usually the pixel coordinates of the set of image points (although this is not the only possible choice), and the parameters $\boldsymbol{a}$ in the definition of $\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{m}, \boldsymbol{a})$ in (34.1) are nothing but the camera intrinsic parameters to go from image measurements expressed in pixels to the features.

34.2.1 The Interaction Matrix

A three-dimensional world point with coordinates $\boldsymbol{X} = (X, Y, Z)$ in the camera frame projects into the image plane of a conventional perspective camera as a two-dimensional point with normalised coordinates $\boldsymbol{x} = (x, y)$. More precisely we have

$$\begin{cases} x &= \dfrac{X}{Z} = \dfrac{u - c_u}{f\alpha} \\ y &= \dfrac{Y}{Z} = \dfrac{v - c_v}{f} \end{cases} , \tag{34.7}$$

where $\boldsymbol{m} = (u, v)$ gives the coordinates of the image point expressed in pixel units, and $\boldsymbol{a} = (c_u, c_v, f, \alpha)$ is the set of camera intrinsic parameters as defined in Chap. 32: c_u and c_v are the coordinates of the principal point, f is the focal length, and α is the ratio of the pixel dimensions. The intrinsic parameter β defined in Chap. 32 has been assumed to be 0 here. In this case, we take $\boldsymbol{s} = \boldsymbol{x} = (x, y)$, the image plane coordinates of the point. The details of the imaging geometry and perspective projection can be found in many computer vision texts, including [34.5–7].

Taking the time derivative of the projection equations (34.7), we obtain

$$\begin{cases} \dot{x} &= \dfrac{\dot{X}}{Z} - \dfrac{X\dot{Z}}{Z^2} = \dfrac{\dot{X} - x\dot{Z}}{Z} \\ \dot{y} &= \dfrac{\dot{Y}}{Z} - \dfrac{Y\dot{Z}}{Z^2} = \dfrac{\dot{Y} - y\dot{Z}}{Z} \end{cases} . \tag{34.8}$$

We can relate the velocity of the 3-D point to the camera spatial velocity using the well-known equation

$$\dot{\boldsymbol{X}} = -\boldsymbol{v}_c - \boldsymbol{\omega}_c \times \boldsymbol{X} \Leftrightarrow \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \,, \end{cases} \tag{34.9}$$

where $\boldsymbol{v}_c = (v_x, v_y, v_z)$ and $\boldsymbol{\omega}_c = (\omega_x, \omega_y, \omega_z)$. Inserting (34.9) into (34.8), grouping terms, and using (34.7) we obtain

$$\begin{cases} \dot{x} = \dfrac{-v_x}{Z} + \dfrac{x v_z}{Z} + xy\omega_x - (1 + x^2)\omega_y + y\omega_z \\ \dot{y} = \dfrac{-v_y}{Z} + \dfrac{y v_z}{Z} + (1 + y^2)\omega_x - xy\omega_y - x\omega_z \,, \end{cases} \tag{34.10}$$

which can be written

$$\dot{\boldsymbol{x}} = \mathbf{L}_{\boldsymbol{x}} \boldsymbol{v}_c \,, \tag{34.11}$$

where the interaction matrix $\mathbf{L}_x$ is given by

$$\mathbf{L}_x = \begin{pmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{pmatrix} . \tag{34.12}$$

In the matrix $\mathbf{L}_x$, the value Z is the depth of the point relative to the camera frame. Therefore, any control scheme that uses this form of the interaction matrix must estimate or approximate the value of Z. Similarly, the camera intrinsic parameters are involved in the computation of x and y. Thus $\mathbf{L}_x^+$ cannot be *directly* used in (34.4), and an estimation or an approximation $\widehat{\mathbf{L}_x^+}$ must be used, as in (34.5). We discuss this in more detail below.

To control the six degrees of freedom, at least three points are necessary (i. e., we require $k \geq 6$). If we use the feature vector $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$, by merely stacking interaction matrices for three points we obtain

$$\mathbf{L}_x = \begin{pmatrix} \mathbf{L}_{\boldsymbol{x}_1} \\ \mathbf{L}_{\boldsymbol{x}_2} \\ \mathbf{L}_{\boldsymbol{x}_3} \end{pmatrix} .$$

In this case, there will exist some configurations for which $\mathbf{L}_x$ is singular [34.8]. Furthermore, there exist four distinct camera poses for which $\boldsymbol{e} = \mathbf{0}$, i. e., four global minima exist for the error function $\|\boldsymbol{e}\|$, and it is impossible to differentiate them [34.9]. For these reasons, more than three points are usually considered.

34.2.2 Approximating the Interaction Matrix

There are several choices available for constructing the estimate $\widehat{\mathbf{L}_e^+}$ to be used in the control law. One popular scheme is of course to choose

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$$

if $\mathbf{L}_e = \mathbf{L}_x$ is known, that is if the current depth Z of each point is available [34.2]. In practice, these parameters must be estimated at each iteration of the control scheme. The basic IBVS methods use classical pose-estimation methods (Chap. 32 and the beginning of Sect. 34.3). Another popular approach is to choose

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$$

where $\mathbf{L}_{e^*}$ is the value of $\mathbf{L}_e$ for the desired position $\boldsymbol{e} = \boldsymbol{e}^* = 0$ [34.3]. In this case, $\widehat{\mathbf{L}_e^+}$ is constant, and only the desired depth of each point has to be set, which means no varying 3-D parameters have to be estimated during the visual servo. Finally, the choice

$$\widehat{\mathbf{L}_e^+} = \left(\frac{\mathbf{L}_e}{2} + \frac{\mathbf{L}_{e^*}}{2} \right)^+$$

has been proposed [34.10]. Since $\mathbf{L}_e$ is involved in this method, the current depth of each point also has to be available.

We illustrate the behavior of these control schemes with an example. The goal is to position the camera so that it observes a square centered in the image (Fig. 34.1). We define $\boldsymbol{s}$ to include the x and y coordinates of the four points forming the square. Note

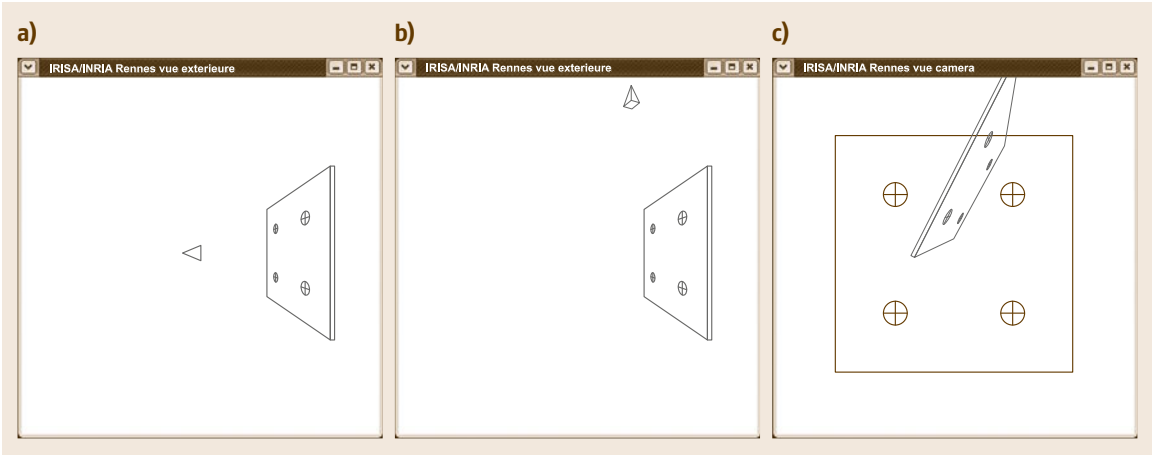


Fig.34.1a–c Example of positioning task: **(a)** the desired camera pose with respect to a simple target, **(b)** the initial camera pose, and **(c)** the corresponding initial and desired image of the target

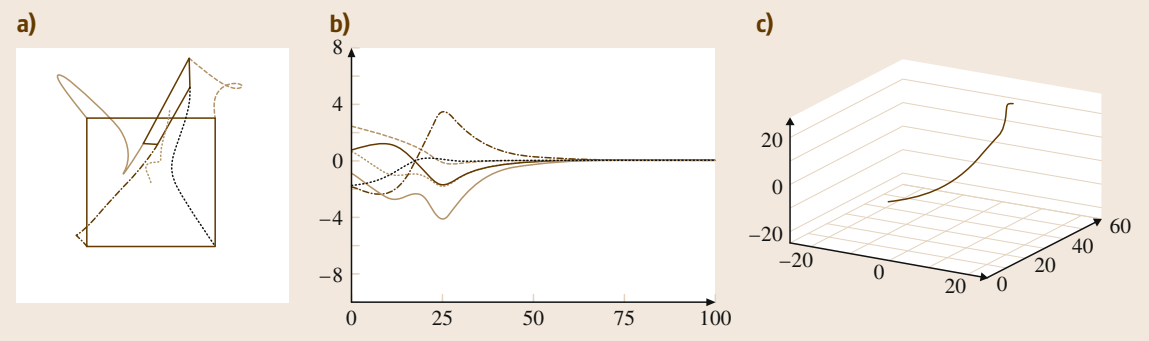


Fig.34.2a–c IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$: **(a)** image point trajectories including the trajectory of the center of the square, which is not used in the control scheme, **(b)** $\boldsymbol{v}_c$ components (cm/s and deg/s) computed at each iteration of the control scheme, and **(c)** the 3-D trajectory of the camera optical center expressed in the desired camera frame $\mathcal{R}_{c^*}$ (cm)

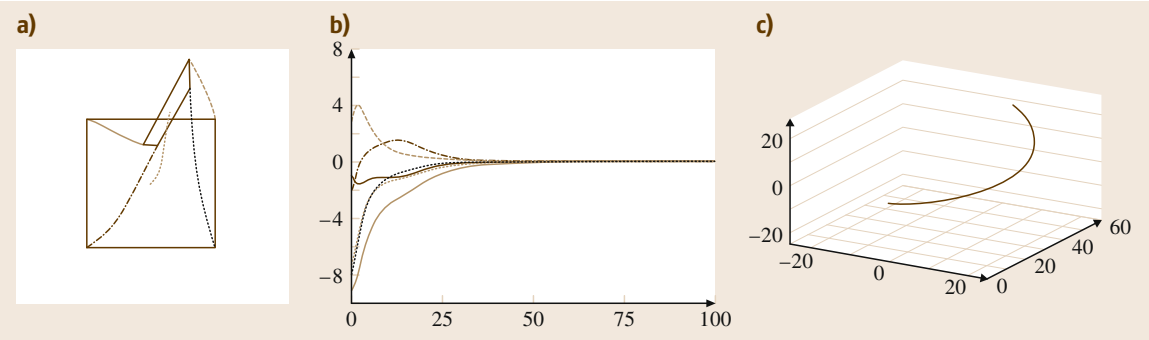


Fig.34.3a–c IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$ (refer to Fig. 34.2a-c for description)

that the initial camera pose has been selected far away from the desired pose, particularly with regard to the rotational motions, which are known to be the most problematic for IBVS. In the simulations presented in the following, no noise or modeling errors have been introduced in order to allow comparison of different behaviors in perfect conditions. The accompanying videos show experimental results obtained using an Adept Viper robot arm and the ViSP library [34.11]. The videos all show the same task being performed from the same initial conditions, and only the control approach varies.

The results obtained by using

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$$

are given in Fig. 34.2 and VIDEO 59. Note that despite the large displacement that is required the system converges. However, neither the behavior in the image, nor the computed camera velocity components, nor the 3-D trajectory of the camera present desirable properties far from the convergence (i. e., for the first 30 or so iterations).

The results obtained using

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$$

are given in Fig. 34.3 and VIDEO 60. In this case, the trajectories of the points in the image are almost straight lines, which means that the points never leave the camera's field of view. However the behavior induced in the camera frame is even less satisfactory than for the case of

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+ .$$

The large camera velocities at the beginning of the servo indicate that the condition number of $\widehat{\mathbf{L}_e^+}$ is high at the start of the trajectory, and the camera trajectory is far from a straight line.

The choice

$$\widehat{\mathbf{L}_e^+} = \left(\frac{\mathbf{L}_e}{2} + \frac{\mathbf{L}_{e^*}}{2}\right)^+$$

provides good performance in practice. Indeed, as can be seen in Fig. 34.4, the camera velocity components do not include large oscillations, which provides

Avidbots Ex1016
Page 879 of 2154

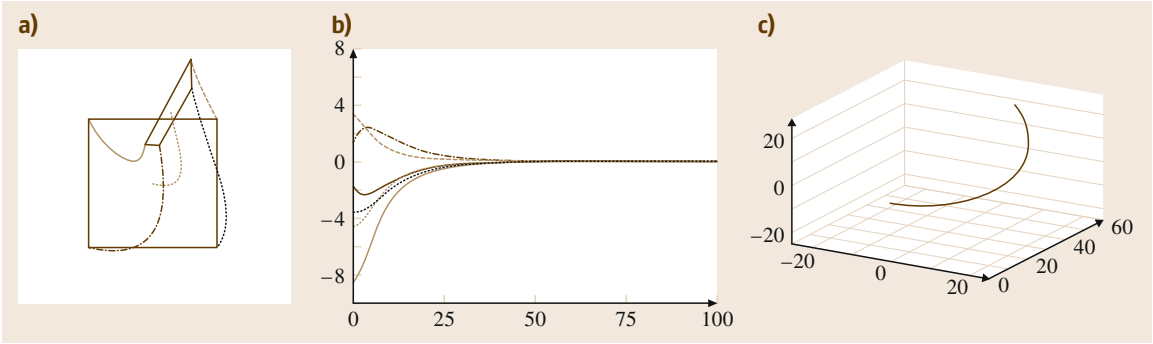


Fig. 34.4 IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \left(\mathbf{L}_e/2 + \mathbf{L}_e^*/2\right)^+$ (refer to Fig. 34.2a-c for description)

a smooth trajectory in both the image and in 3-D space (VIDEO 61).

34.2.3 A Geometrical Interpretation of IBVS

It is quite easy to provide a geometric interpretation of the behavior of the control schemes defined above. The example illustrated in Fig. 34.5 corresponds to a pure rotation around the optical axis from the initial configuration (shown in blue) to the desired configuration of four coplanar points parallel to the image plane (shown in red).

As explained above, using $\mathbf{L}_e^+$ in the control scheme attempts to ensure an exponential decrease of the error $\boldsymbol{e}$. This means that, when x and y image point coordinates compose this error, the points' trajectories in the image follow straight lines from their initial to their desired positions, when this is possible. This leads to the image motion plotted in green in the figure. The camera motion to realize this image motion can be easily deduced and is indeed composed of a rotational motion around the optical axis, but combined with a retreating translational motion along the optical axis [34.12]. This unexpected motion is due to the choice of the features and the form of the third and sixth columns in the interaction matrix, which induces a coupling between the features and the two degrees of freedom involved (v_z and ω_z). If the rotation between the initial and desired configurations is very large, this phenomenon is amplified, and leads to a particular case for a rotation of π radians where no rotational motion at all will be induced by the control scheme [34.13]. On the other hand, when the rotation is small, this phenomenon almost disappears. To conclude, the behavior is locally satisfactory (i. e., when the error is small), but it can be unsatisfactory when the error is large. As we will see below, these results are consistent with the local asymptotic stability results that can be obtained for IBVS.

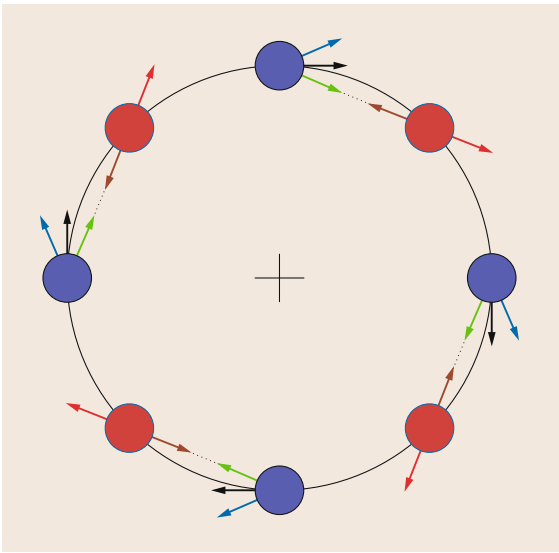


Fig. 34.5 Geometrical interpretation of IBVS: going from the *blue position* to the *red one*. In *green*, image motion when $\mathbf{L}_e^+$ is used in the control scheme; in *blue*, when $\mathbf{L}_{e^*}^+$ is used; and in *black* when $(\mathbf{L}_e/2 + \mathbf{L}_{e^*}/2)^+$ is used (see the text for more details)

If instead we use $\mathbf{L}_{e^*}^+$ in the control scheme, the image motion generated can easily be shown to be the blue one plotted in Fig. 34.5. Indeed, if we consider the same control scheme as before but starting from $\boldsymbol{s}^*$ to reach $\boldsymbol{s}$, we obtain

$$\boldsymbol{v}_c = -\lambda \mathbf{L}_{e^*}^+ (\boldsymbol{s}^* - \boldsymbol{s}) \,,$$

which again induces straight-line trajectories from the red points to the blue ones, causing the image motion plotted in brown. Going back to our problem, the control scheme computes a camera velocity that is exactly the opposite one

$$\boldsymbol{v}_c = -\lambda \mathbf{L}_{e^*}^+ (\boldsymbol{s} - \boldsymbol{s}^*) \,,$$

and thus generates the image motion plotted in red at the red points. Transformed to the blue points, the camera velocity generates the blue image motion and corresponds once again to a rotational motion around the optical axis, combined now with an unexpected forward motion along the optical axis. The same analysis can be done as before for the case of large or small errors. We can add that, as soon as the error decreases significantly, both control schemes get closer, and tend to the same one (since $\mathbf{L}_e = \mathbf{L}_{e^*}$ when $\boldsymbol{e} = \boldsymbol{e}^*$) with a nice behavior characterized with the image motion plotted in black and a camera motion composed of only a rotation around the optical axis when the error tends towards zero.

If we instead use

$$\widehat{\mathbf{L}_e^+} = \left(\frac{\mathbf{L}_e}{2} + \frac{\mathbf{L}_{e^*}}{2}\right)^+ ,$$

it is intuitively clear that considering the mean of $\mathbf{L}_e$ and $\mathbf{L}_{e^*}$ generates the image motion plotted in black, even when the error is large. In all cases but the rotation of π radians, the camera motion is now a pure rotation around the optical axis, without any unexpected translational motion.

34.2.4 Stability Analysis

We now consider the fundamental issues related to the stability of IBVS. To assess the stability of the closed-loop visual servo systems, we will use Lyapunov analysis. In particular, consider the candidate Lyapunov function defined by the squared error norm $\mathcal{L} = \frac{1}{2}\|\boldsymbol{e}(t)\|^2$, whose derivative is given by

$$\dot{\mathcal{L}} = \boldsymbol{e}^\mathrm{T}\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}^\mathrm{T}\mathbf{L}_e\widehat{\mathbf{L}_e^+}\boldsymbol{e}$$

since $\dot{\boldsymbol{e}}$ is given by (34.6). The global asymptotic stability of the system is thus obtained when the following sufficient condition is satisfied

$$\mathbf{L}_e\widehat{\mathbf{L}_e^+} > 0\,. \tag{34.13}$$

If the number of features is equal to the number of camera degrees of freedom (i. e., $k = 6$), and if the features are chosen and the control scheme designed so that $\mathbf{L}_e$ and $\widehat{\mathbf{L}_e^+}$ are of full rank 6, then the condition (34.13) is satisfied if the approximations involved in $\widehat{\mathbf{L}_e^+}$ are not too coarse.

As discussed above, for most IBVS approaches we have $k > 6$. Therefore the condition (34.13) can never be ensured since

$$\mathbf{L}_e\widehat{\mathbf{L}_e^+} \in \mathbb{R}^{k\times k}$$

is at most of rank 6, and thus $\mathbf{L}_e\widehat{\mathbf{L}_e^+}$ has a nontrivial null space. In this case, configurations such that $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_e^+}$ correspond to local minima. Reaching such a local minimum is illustrated in Fig. 34.6. As can be seen in Fig. 34.6d, each component of $\boldsymbol{e}$ has a nice exponential decrease with the same convergence speed, causing straight-line trajectories to be realized in the image, but the error reached is not exactly zero, and it is clear from Fig. 34.6c that the system has been attracted to a local minimum far away from the desired configuration. Thus, only local asymptotic stability can be obtained for IBVS.

To study local asymptotic stability when $k > 6$, let us first define a new error $\boldsymbol{e}'$ with $\boldsymbol{e}' = \widehat{\mathbf{L}_e^+}\boldsymbol{e}$. The time derivative of this error is given by

$$\dot{\boldsymbol{e}}' = \widehat{\mathbf{L}_e^+}\dot{\boldsymbol{e}} + \dot{\widehat{\mathbf{L}_e^+}}\boldsymbol{e} = \left(\widehat{\mathbf{L}_e^+}\mathbf{L}_e + \boldsymbol{O}\right)\boldsymbol{v}_\mathrm{c}\,,$$

where $\boldsymbol{O} \in \mathbb{R}^{6\times 6}$ is equal to $\mathbf{0}$ when $\boldsymbol{e} = 0$, whatever the choice of $\widehat{\mathbf{L}_e^+}$ [34.14]. Using the control scheme (34.5),w e obtain

$$\dot{\boldsymbol{e}}' = -\lambda\left(\widehat{\mathbf{L}_e^+}\mathbf{L}_e + \boldsymbol{O}\right)\boldsymbol{e}'\,,$$

which is known to be locally asymptotically stable in a neighborhood of $\boldsymbol{e} = \boldsymbol{e}^* = \mathbf{0}$ if

$$\widehat{\mathbf{L}_e^+}\mathbf{L}_e > 0\,, \tag{34.14}$$

where $\widehat{\mathbf{L}_e^+}\mathbf{L}_e \in \mathbb{R}^{6\times 6}$. Indeed, only the linearized system

$$\dot{\boldsymbol{e}}' = -\lambda\widehat{\mathbf{L}_e^+}\mathbf{L}_e\boldsymbol{e}'$$

has to be considered if we are interested in the local asymptotic stability [34.15].

Once again, if the features are chosen and the control scheme designed so that $\mathbf{L}_e$ and $\widehat{\mathbf{L}_e^+}$ are of full rank 6, then condition (34.14) is ensured if the approximations involved in $\widehat{\mathbf{L}_e^+}$ are not too coarse.

To end the demonstration of local asymptotic stability, we must show that there does not exist any configuration $\boldsymbol{e} \neq \boldsymbol{e}^*$ such that $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_e^+}$ in a small neighborhood of $\boldsymbol{e}^*$ and in a small neighborhood of the corresponding pose $\boldsymbol{p}^*$. Such configurations correspond to local minima where $\boldsymbol{v}_\mathrm{c} = 0$ and $\boldsymbol{e} \neq \boldsymbol{e}^*$. If such a pose $\boldsymbol{p}$ would exist, it is possible to restrict the neighborhood around $\boldsymbol{p}^*$ so that there exists a camera velocity $\boldsymbol{v}$ to reach $\boldsymbol{p}^*$ from $\boldsymbol{p}$. This camera velocity would imply a variation of the error $\dot{\boldsymbol{e}} = \mathbf{L}_e\boldsymbol{v}$. However, such a variation cannot belong to $\ker \widehat{\mathbf{L}_e^+}$ since $\widehat{\mathbf{L}_e^+}\mathbf{L}_e > 0$. Therefore, we have $\boldsymbol{v}_\mathrm{c} = 0$ if and only if $\dot{\boldsymbol{e}} = \mathbf{0}$, i. e., $\boldsymbol{e} = \boldsymbol{e}^*$, in a neighborhood of $\boldsymbol{p}^*$.

Avidbots Ex1016
Page 881 of 2154

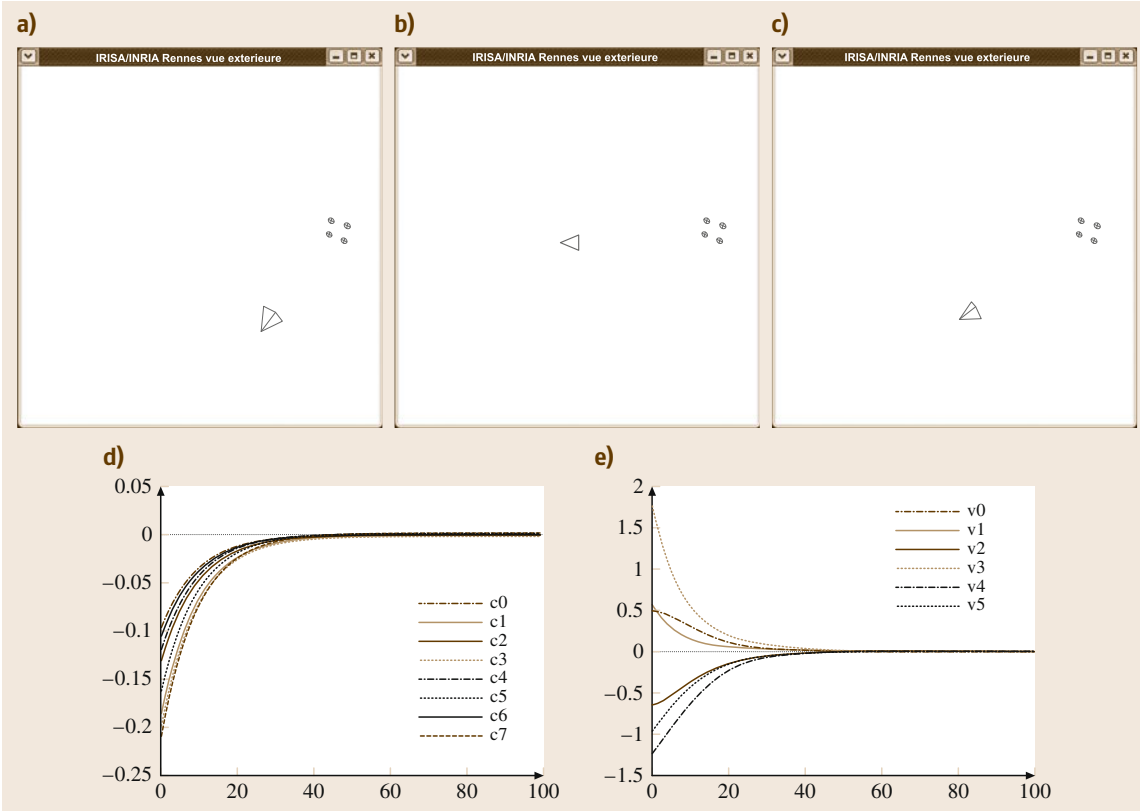


Fig.34.6a–e IBVS reaching a local minimum using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}}^{+}$: **(a)** the initial configuration, **(b)** the desired one, **(c)** the configuration reached after the convergence of the control scheme, **(d)** the evolution of the error $\boldsymbol{e}$ at each iteration of the control scheme, and **(e)** the evolution of the six components of the camera velocity $\boldsymbol{v}_c$

Even though local asymptotic stability can be ensured when $k > 6$, we recall that global asymptotic stability cannot be ensured. For instance, as illustrated in Fig. 34.6, there may exist local minima corresponding to configurations where $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$, which are outside of the neighborhood considered above. Determining the size of the neighborhood in which stability and the convergence are ensured is still an open issue, even if this neighborhood is surprisingly quite large in practice.

34.2.5 IBVS with a Stereo Vision System

It is straightforward to extend the IBVS approach to a multicamera system. If a stereo vision system is used, and a world point is visible in both left and right images (Fig. 34.7), it is possible to use as visual features

$$\boldsymbol{s} = \boldsymbol{x}_s = (\boldsymbol{x}_l, \boldsymbol{x}_r) = (x_l, y_l, x_r, y_r)$$

i. e., to represent the point by just stacking in $\boldsymbol{s}$ the x and y coordinates of the observed point in the left and right images [34.16]. However, care must be taken when constructing the corresponding interaction matrix since the form given in (34.11) is expressed in either the left or right camera frame. More precisely, we have

$$\begin{cases} \dot{\boldsymbol{x}}_l &= \mathbf{L}_{\boldsymbol{x}_l}\, \boldsymbol{v}_l \,, \\ \dot{\boldsymbol{x}}_r &= \mathbf{L}_{\boldsymbol{x}_r}\, \boldsymbol{v}_r \,, \end{cases}$$

where $\boldsymbol{v}_l$ and $\boldsymbol{v}_r$ are the spatial velocity of the left and right camera, respectively, and where the analytical form of $\mathbf{L}_{\boldsymbol{x}_l}$ and $\mathbf{L}_{\boldsymbol{x}_r}$ are given by (34.12).

By choosing a sensor frame rigidly linked to the stereo vision system, we obtain

$$\dot{\boldsymbol{x}}_s = \begin{pmatrix} \dot{\boldsymbol{x}}_l \\ \dot{\boldsymbol{x}}_r \end{pmatrix} = \mathbf{L}_{\boldsymbol{x}_s}\, \boldsymbol{v}_s \,,$$

where the interaction matrix related to $\boldsymbol{x}_s$ can be determined using the spatial motion transform matrix $\mathbf{V}$ defined in Chap. 02 to transform velocities expressed in the left or right cameras frames to the sensor frame. We

Avidbots Ex1016
Page 882 of 2154

recall that $\mathbf{V}$ is given by

$$\mathbf{V} = \begin{pmatrix} \mathbf{R} & [\boldsymbol{t}]_\times \mathbf{R} \\ \mathbf{0} & \mathbf{R} \end{pmatrix} , \tag{34.15}$$

where $[\boldsymbol{t}]_\times$ is the skew-symmetric matrix associated to the vector $\boldsymbol{t}$ and where $(\mathbf{R}, \boldsymbol{t}) \in SE(3)$ is the rigid-body transformation from the camera to the sensor frame. The numerical values for these matrices are directly obtained from the calibration step of the stereo vision system. Using this equation, we obtain

$$\mathbf{L}_{x_s} = \begin{pmatrix} \mathbf{L}_{x_l}\, {}^{l}\mathbf{V}_s \\ \mathbf{L}_{x_r}\, {}^{r}\mathbf{V}_s \end{pmatrix} .$$

Note that $\mathbf{L}_{x_s} \in \mathbb{R}^{4\times 6}$ is always of rank 3 because of the epipolar constraint that links the perspective projection of a 3-D point in a stereo vision system (Fig. 34.7). Another simple interpretation is that a 3-D point is represented by three independent parameters, which makes it impossible to find more than three independent parameters using any sensor observing that point.

To control the six degrees of freedom of the system, it is necessary to consider at least three points, as the rank of the interaction matrix considering only two points is 5.

Using a stereo vision system, since the 3-D coordinates of any point observed in both images can be easily estimated by a simple triangulation process it is possible and quite natural to use these 3-D coordinates in the features set $\boldsymbol{s}$. Such an approach would be, strictly speaking, a position-based approach, since it would require 3-D parameters in $\boldsymbol{s}$.

34.2.6 IBVS with Cylindrical Coordinates of Image Points

In the previous sections, we have considered the Cartesian coordinates of image points. As proposed in [34.17] it may be useful to consider instead the cylindrical coordinates $\boldsymbol{\gamma} = (\rho, \theta)$ of the image points instead of their Cartesian coordinates $\boldsymbol{x} = (x, y)$. They are given by

$$\rho = \sqrt{x^2 + y^2} \ , \ \theta = \arctan \frac{y}{x}$$

from which we deduce

$$\dot{\rho} = \frac{x\dot{x} + y\dot{y}}{\rho} \ , \ \dot{\theta} = \frac{x\dot{y} - y\dot{x}}{\rho^2} \ .$$

Using (34.11) and then substituting x by $\rho\cos\theta$ and y by $\rho\sin\theta$, we obtain immediately

$$\mathbf{L}_{\boldsymbol{\gamma}} = \begin{pmatrix} \frac{-c}{Z} & \frac{-s}{Z} & \frac{\rho}{Z} & (1+\rho^2)s & -(1+\rho^2)c & 0 \\ \frac{s}{\rho Z} & \frac{-c}{\rho Z} & 0 & \frac{c}{\rho} & \frac{s}{\rho} & -1 \end{pmatrix} , \tag{34.16}$$

where $c = \cos\theta$ and $s = \sin\theta$. Note that θ is not defined when the image point lies at the principal point (where $x = y = \rho = 0$). It is thus not surprising that the interaction matrix $\mathbf{L}_{\boldsymbol{\gamma}}$ is singular in that case.

If we go back to the example depicted in Fig. 34.5, the behavior obtained using cylindrical coordinates will be the expected one, that is a pure rotation around the optical axis, by using either $\mathbf{L}_{e}^{+}$, $\mathbf{L}_{e^*}^{+}$ or $(\mathbf{L}_e/2 + \mathbf{L}_{e^*}/2)^{+}$ in the control scheme. This is due to the form of the third and sixth columns of the interaction matrix (34.16), which leads to a decoupled system.

34.2.7 IBVS with Other Geometrical Features

In the previous sections, we have only considered image point coordinates in $\boldsymbol{s}$. Other geometrical primitives can of course be used. There are several reasons to do so. Firstly, the scene observed by the camera cannot always be described merely by a collection of points, in which case the image processing provides other types of measurements, such as a set of straight lines or the contours of an object. Secondly, richer geometric primitives may ameliorate the decoupling and linearizing issues that motivate the design of partitioned systems (Sect. 34.4). Finally, the robotic task to be achieved may be expressed in terms of virtual linkages (or fixtures) between the camera and the observed objects [34.18, 19], sometimes expressed directly by constraints between primitives, such as point-to-line [34.20] (which means that an observed point must lie on a specified line).

It is possible to determine the interaction matrix related to the perspective projection of a large class of geometrical primitives, such as segments, straight lines, spheres, circles, and cylinders. The results are given in [34.3, 18]. Recently, the analytical form of the interaction matrix related to any image moments corresponding to planar objects has been computed. This

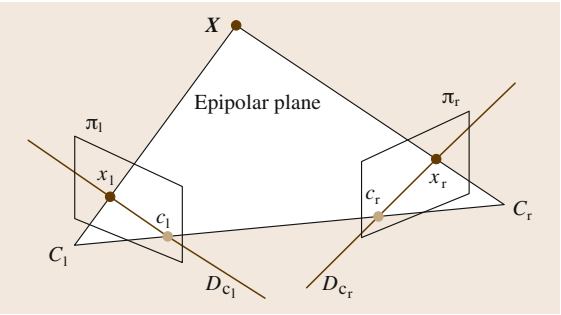


Fig. 34.7 A stereo vision system

Avidbots Ex1016
Page 883 of 2154

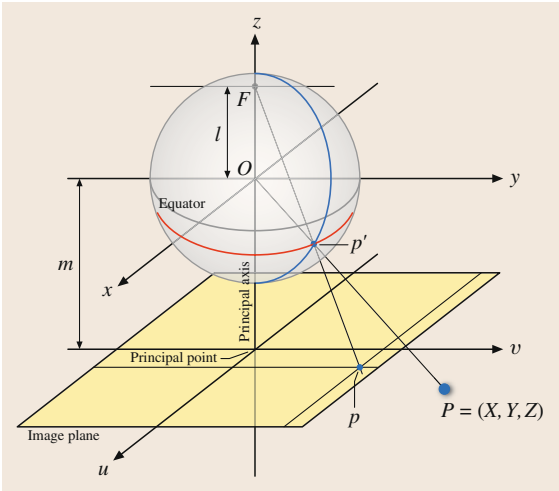


Fig. 34.8 Unified imaging model of Geyer and Daniilidis (after [34.7])

makes it possible to consider planar objects of any shape [34.21]. If a collection of points is measured in the image, moments can also be used [34.22]. In both cases, moments allow the use of intuitive geometrical features, such as the center of gravity or the orientation of an object. By selecting an adequate combination of moments, it is then possible to determine partitioned systems with good decoupling and linearizing properties [34.21, 22].

Note that, for all these features (geometrical primitives, moments), the depth of the primitive or of the object considered appears in the coefficients of the interaction matrix related to the translational degrees of freedom, as was the case for the image points. An estimation of this depth is thus generally necessary (Sect. 34.6). In a few situations, for instance with a suitable normalization of moments [34.22], only the constant desired depth appears in the interaction matrix, which makes estimating depth unnecessary.

34.2.8 Non-Perspective Cameras

The vast majority of cameras we use, and our own eyes, are characterized by a perspective projection model which closely approximate the ideal pinhole imaging model. Such cameras have a narrow field of view, typically less than one half hemisphere. For robotics it is often advantageous to have a large field of view and this can be achieved using a fisheye lens camera or a catadioptric (lens and mirror system) camera (often referred to as a panoramic camera). For these non-perspective sensors the interaction matrix of any visual feature has a different form to those discussed above, such as (34.12) and (34.16) for an image point.

Rather than determine the interaction matrix of visual features expressed in the image plane of non-perspective cameras, we can transform the images from these cameras to the view that would be seen by an ideal spherical camera. The spherical model projects world points onto a unit sphere, the intersection of the unit sphere with the ray from the world point to the center of the sphere. Such an ideal camera has the largest possible field of view. The unified imaging model [34.23] shown in Fig 34.8b provides a general mechanism to project a world point to the image plane of a large class of cameras. To be precise it includes all central projection cameras and this includes perspective and some catadioptric cameras with particular mirror shapes, but in practice it is a very good approximation to non-central cameras including fisheye and general catadioptric systems. The mechanism of this unified model can also be used to reproject points from these varied image planes to a spherical camera.

Referring to Fig. 34.8 the world point **P** is represented by the vector $\boldsymbol{X} = (X, Y, Z)$ in the camera frame, and is projected onto the surface of the unit sphere at the point $\boldsymbol{x}_s = (x_s, y_s, z_s)$ with

$$x_s = \frac{X}{R}\,, \quad y_s = \frac{Y}{R}\,, \quad \text{and} \quad z_s = \frac{Z}{R}\,,$$

where $R = \sqrt{X^2 + Y^2 + Z^2}$ is the distance from the sphere center to the world point.

The interaction matrix of $\boldsymbol{x}_s$ is derived in essentially the same manner as for the perspective camera, and it can be shown to be [34.24]

$$\mathbf{L}_{\boldsymbol{x}_s} = \left[\frac{1}{R}\left(\boldsymbol{x}_s \boldsymbol{x}_s{}^{\mathrm{T}} - \mathbf{I}_3\right)[\boldsymbol{x}_s]_{\times}\right]\,. \tag{34.17}$$

Note that R can be expressed as a function of the point depth Z by using $R = Z\sqrt{1 + x_s^2 + y_s^2}$. Therefore, the general spherical model does not add any supplementary unknown in the interaction matrix.

A particular advantage of the spherical model is that for pure camera rotation the shape of an object is invariant, which eases determining visual features that are only linked to translational motions.

34.2.9 Direct Estimation

In the previous sections, we have focused on the analytical form of the interaction matrix. It is also possible to estimate its numerical value directly using either an offline learning step, or an online estimation scheme.

All the methods proposed to estimate the interaction matrix numerically rely on the observation of a variation of the features due to a known or measured camera motion. More precisely, if we measure a feature's variation $\Delta \boldsymbol{s}$ due to a camera motion $\Delta \boldsymbol{v}_c$, we

have from (34.2)

$$\mathbf{L}_s \Delta \boldsymbol{v}_c = \Delta \boldsymbol{s} \,,$$

which provides k equations while we have $k \times 6$ unknown values in $\mathbf{L}_s$. Using a set of N independent camera motions with $N > 6$, it is thus possible to estimate $\mathbf{L}_s$ by solving

$$\mathbf{L}_s \boldsymbol{A} = \boldsymbol{B} \,,$$

where the columns of $\boldsymbol{A} \in \mathbb{R}^{6\times N}$ and $\boldsymbol{B} \in \mathbb{R}^{k\times N}$ are, respectively, formed from the set of camera motions and the set of corresponding features variations. The least-square solution is of course given by

$$\widehat{\mathbf{L}_s} = \boldsymbol{B}\boldsymbol{A}^{+} \,. \tag{34.18}$$

Methods based on neural networks have also been developed to estimate $\mathbf{L}_s$ [34.25, 26]. It is also possible to estimate the numerical value of $\mathbf{L}_s^{+}$ directly, which in practice provides a better behavior [34.27]. In this case, the basic relation is

$$\mathbf{L}_s^{+} \Delta \boldsymbol{s} = \Delta \boldsymbol{v}_c \,,$$

which provides six equations. Using a set of N measurements, with $N > k$, we now obtain

$$\widehat{\mathbf{L}_s^{+}} = \boldsymbol{A}\boldsymbol{B}^{+} \,. \tag{34.19}$$

In the first case (34.18), the six columns of $\mathbf{L}_s$ are estimated by solving six linear systems, while in the second case (34.19), the k columns of $\mathbf{L}_s^{+}$ are estimated by solving k linear systems, which explains the difference in the results.

Estimating the interaction matrix online can be viewed as an optimization problem, and consequently a number of researchers have investigated approaches that derive from optimization methods. These methods typically discretize the system equation (34.2), and use an iterative updating scheme to refine the estimate of $\widehat{\mathbf{L}_s}$ at each stage. One such online and iterative formulation uses the Broyden update rule given by [34.28, 29]

$$\widehat{\mathbf{L}_s}(t+1) = \widehat{\mathbf{L}_s}(t) + \frac{\alpha}{\Delta \boldsymbol{v}_c^{\mathrm{T}} \Delta \boldsymbol{v}_c} \left[\Delta \boldsymbol{x} - \widehat{\mathbf{L}_s}(t) \Delta \boldsymbol{v}_c \right] \Delta \boldsymbol{v}_c^{\mathrm{T}} \,,$$

where α defines the update speed. This method has been generalized to the case of moving objects in [34.30].

The main interest of using such numerical estimations in the control scheme is that it avoids all the modeling and calibration steps. It is particularly useful when using features whose interaction matrix is not available in analytical form. For instance, in [34.31], the main eigenvalues of the principal component analysis of an image have been considered in a visual servoing scheme. The drawback of these methods is that no theoretical stability and robustness analysis can be made.

34.3 Pose-Based Visual Servo

Pose-based control schemes (PBVS) [34.1, 32, 33] use the pose of the camera with respect to some reference coordinate frame to define $\boldsymbol{s}$. Computing this pose from a set of measurements in one image necessitates the camera intrinsic parameters and the 3-D model of the object observed to be known. This classic computer vision problem is called the 3-D localization problem. While this problem is beyond the scope of the present chapter, many solutions have been presented in the literature [34.34, 35] and its basic principles are recalled in Chap. 32.

It is then typical to define $\boldsymbol{s}$ in terms of the parameterization used to represent the camera pose. Note that the parameters $\boldsymbol{a}$ involved in the definition (34.1) of $\boldsymbol{s}$ are now the camera intrinsic parameters and the 3-D model of the object.

It is convenient to consider three coordinate frames: the current camera frame $\mathcal{F}_c$, the desired camera frame $\mathcal{F}_{c^*}$, and a reference frame $\mathcal{F}_o$ attached to the object. We adopt here the standard notation of using a leading superscript to denote the frame with respect to which a set of coordinates is defined. Thus, the coordinate vectors ${}^{c}\boldsymbol{t}_o$ and ${}^{c^*}\boldsymbol{t}_o$ give the coordinates of the origin of the object frame expressed relative to the current camera frame, and relative to the desired camera frame, respectively. Furthermore, let $\mathbf{R} = {}^{c^*}\mathbf{R}_c$ be the rotation matrix that gives the orientation of the current camera frame relative to the desired frame.

We can define $\boldsymbol{s}$ to be $(\boldsymbol{t}, \theta\boldsymbol{u})$, in which $\boldsymbol{t}$ is a translation vector, and $\theta\boldsymbol{u}$ gives the angle/axis parameterization for the rotation. We now discuss two choices for $\boldsymbol{t}$, and give the corresponding control laws.

If $\boldsymbol{t}$ is defined relative to the object frame $\mathcal{F}_o$, we obtain

$$\boldsymbol{s} = ({}^{c}\boldsymbol{t}_o, \theta\boldsymbol{u}) \,, \boldsymbol{s}^* = ({}^{c^*}\boldsymbol{t}_o, 0) \,,$$

and

$$\boldsymbol{e} = ({}^{c}\boldsymbol{t}_o - {}^{c^*}\boldsymbol{t}_o, \theta\boldsymbol{u}) \,.$$

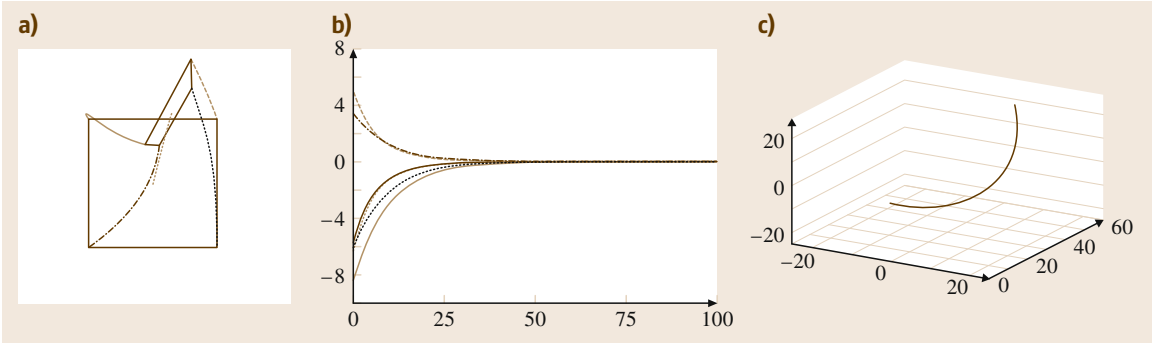


Fig. 34.9 PBVS system behavior using $\boldsymbol{s} = ({}^{c}\boldsymbol{t}_{o}, \theta\boldsymbol{u})$ (refer to Fig. 34.2a-c for description)

In this case, the interaction matrix related to $\boldsymbol{e}$ is given by

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} -\mathbf{I}_3 & [{}^{c}\boldsymbol{t}_{o}]_{\times} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} , \tag{34.20}$$

in which $\mathbf{I}_3$ is the 3×3 identity matrix and $\mathbf{L}_{\theta\boldsymbol{u}}$ is given by [34.36]

$$\mathbf{L}_{\theta\boldsymbol{u}} = \mathbf{I}_3 + \frac{\theta}{2}[\boldsymbol{u}]_{\times} + \left(1 - \frac{\operatorname{sinc}\theta}{\operatorname{sinc}^2\theta/2}\right)[\boldsymbol{u}]_{\times}^2 , \tag{34.21}$$

where $\operatorname{sinc} x$ is the sinus cardinal defined such that $x \operatorname{sinc} x = \sin x$ and $\operatorname{sinc} 0 = 1$.

Following the development in Sect. 34.1, we obtain the control scheme

$$\boldsymbol{v}_{c} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} \boldsymbol{e}$$

since the dimension k of $\boldsymbol{s}$ is six, that is, the number of camera degrees of freedom. By setting

$$\widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} = \begin{pmatrix} -\mathbf{I}_3 & [{}^{c}\boldsymbol{t}_{o}]_{\times}\mathbf{L}_{\theta\boldsymbol{u}}^{-1} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}}^{-1} \end{pmatrix} , \tag{34.22}$$

we obtain after simple developments

$$\begin{cases} \boldsymbol{v}_{c} = -\lambda\left[{}^{c^*}\boldsymbol{t}_{o} - {}^{c}\boldsymbol{t}_{o}\right] + [{}^{c}\boldsymbol{t}_{o}]_{\times}\,\theta\boldsymbol{u} \\ \boldsymbol{\omega}_{c} = -\lambda\theta\boldsymbol{u} \end{cases} . \tag{34.23}$$

since $\mathbf{L}_{\theta\boldsymbol{u}}$ is such that

$$\mathbf{L}_{\theta\boldsymbol{u}}^{-1}\theta\boldsymbol{u} = \theta\boldsymbol{u} .$$

Ideally, that is, if the pose parameters are perfectly estimated, the behavior of $\boldsymbol{e}$ will be the expected one ($\dot{\boldsymbol{e}} = -\lambda\boldsymbol{e}$). The choice of $\boldsymbol{e}$ causes the rotational motion to follow a geodesic with an exponential decreasing speed and causes the translational parameters involved in $\boldsymbol{s}$ to decrease at the same speed. This explains the nice exponential decrease of the camera velocity components in Fig. 34.9. Furthermore, the trajectory in the image of the origin of the object frame follows a straight line (here the center of the four points has been selected as this origin). On the other hand, the camera trajectory does not follow a straight line (VIDEO 62).

Another PBVS scheme can be designed by using $\boldsymbol{s} = ({}^{c^*}\boldsymbol{t}_{c}, \theta\boldsymbol{u})$. In this case, we have $\boldsymbol{s}^* = \mathbf{0}$, $\boldsymbol{e} = \boldsymbol{s}$, and

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} . \tag{34.24}$$

Note the decoupling between translational and rotational motions, which allows us to obtain a simple control scheme

$$\begin{cases} \boldsymbol{v}_{c} &= -\lambda\mathbf{R}^{\mathrm{T}}\,{}^{c^*}\boldsymbol{t}_{c} \\ \boldsymbol{\omega}_{c} &= -\lambda\,\theta\boldsymbol{u} \end{cases} . \tag{34.25}$$

In this case, as can be seen in Fig. 34.10 and in VIDEO 63, if the pose parameters involved in (34.25) are estimated perfectly, the camera trajectory is a straight line, while the image trajectories are less satisfactory than before. Some particular configurations can be found which will lead to some points leaving the camera field of view during the robot motion.

The stability properties of PBVS seem quite attractive. Since $\mathbf{L}_{\theta\boldsymbol{u}}$ given in (34.21) is nonsingular when $\theta \neq 2k\pi$, $\forall k \in Z^*$, we obtain from (34.13) the global asymptotic stability of the system since $\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} = \mathbf{I}_6$, under the strong hypothesis that all the pose parameters are perfect. This is true for both methods presented above, since the interaction matrices given in (34.20) and (34.24) are full rank when $\mathbf{L}_{\theta\boldsymbol{u}}$ is nonsingular.

Avidbots Ex1016
Page 886 of 2154

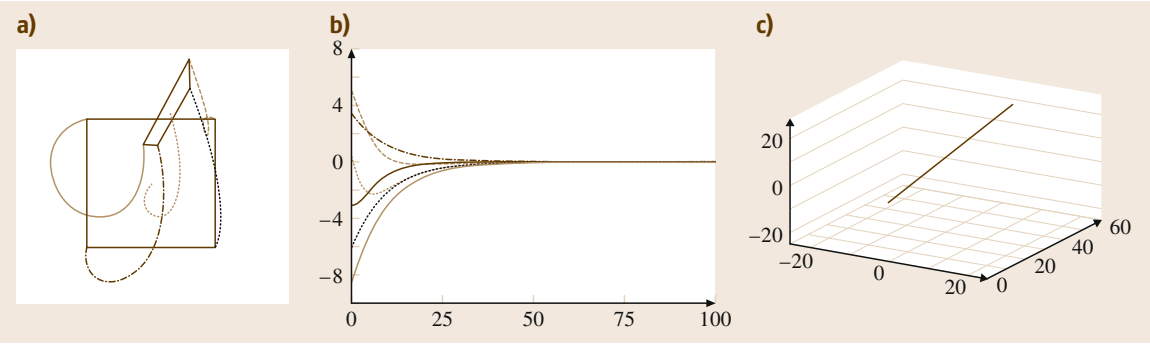


Fig. 34.10 PBVS system behavior using $\boldsymbol{s} = ({}^{c^*}\boldsymbol{t}_c, \theta\boldsymbol{u})$ (refer to Fig. 34.2a-c for description)

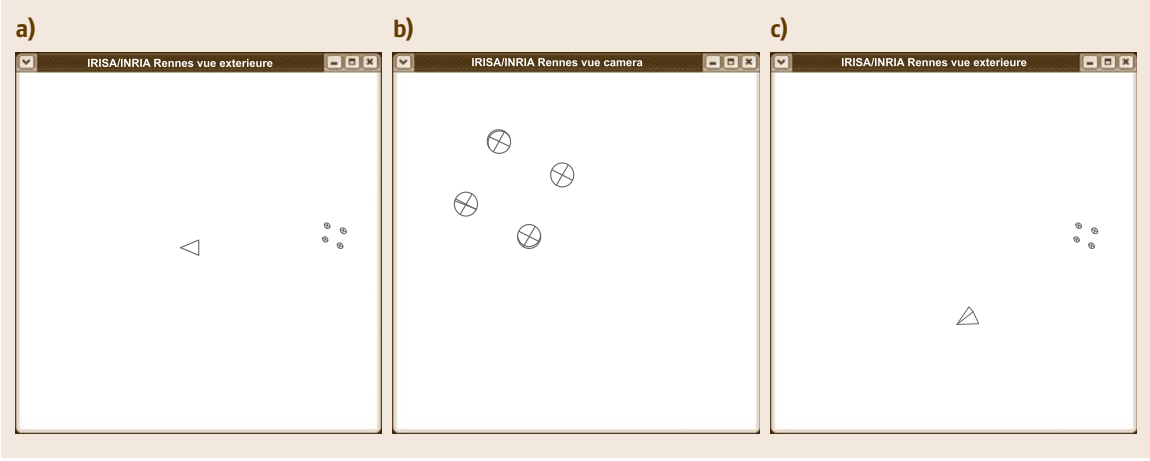


Fig.34.11a–c Two different camera poses **(a,c)** that provide almost the same image of four coplanar points shown overlaid in **(b)**

With regard to robustness, feedback is computed using *estimated* quantities that are a function of the image measurements and the system calibration parameters. For the first method presented in Sect. 34.3 (the analysis for the second method is analogous), the interaction matrix given in (34.20) corresponds to perfectly estimated pose parameters, while the real one is unknown since the estimated pose parameters may be biased due to calibration errors, or inaccurate and unstable due to noise [34.13]. The true positivity condition (34.13) should in fact be written

$$\mathbf{L}_{\hat{e}}\widehat{\mathbf{L}_{\hat{e}}^{-1}} > 0\,, \tag{34.26}$$

where $\widehat{\mathbf{L}_{\hat{e}}^{-1}}$ is given by (34.22) but where $\mathbf{L}_{\hat{e}}$ is unknown, and not given by (34.20). Indeed, even small errors in computing the position of points in the image can lead to pose errors which will significantly impact the accuracy and the stability of the system (Fig. 34.11).

Avidbots Ex1016
Page 887 of 2154

34.4 Advanced Approaches

34.4.1 Hybrid VS

Suppose we have access to a control law for $\boldsymbol{\omega}_c$, such as the one used in PBVS ((34.23) or (34.25))

$$\boldsymbol{\omega}_c = -\lambda\, \theta \boldsymbol{u}\,. \tag{34.27}$$

How could we use this in conjunction with IBVS?

Considering a feature vector $\boldsymbol{s}_t$ and an error $\boldsymbol{e}_t$ devoted to control the translational degrees of freedom, we can partition the interaction matrix as follows

$$\begin{aligned}\dot{\boldsymbol{s}}_t &= \mathbf{L}_{\boldsymbol{s}_t} \boldsymbol{v}_c \\ &= \begin{pmatrix}\mathbf{L}_v & \mathbf{L}_{\boldsymbol{\omega}}\end{pmatrix}\begin{pmatrix}\boldsymbol{v}_c \\ \boldsymbol{\omega}_c\end{pmatrix} \\ &= \mathbf{L}_v \boldsymbol{v}_c + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c\,.\end{aligned}$$

Now, setting $\dot{\boldsymbol{e}}_t = -\lambda \boldsymbol{e}_t$, we can solve for the desired translational control input as

$$\begin{aligned}-\lambda \boldsymbol{e}_t = \dot{\boldsymbol{e}}_t = \dot{\boldsymbol{s}}_t = \mathbf{L}_v \boldsymbol{v}_c + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c\,, \\ \Rightarrow \boldsymbol{v}_c = -\mathbf{L}_v^{+}(\lambda \boldsymbol{e}_t + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c)\,.\end{aligned} \tag{34.28}$$

We can think of the quantity $(\lambda \boldsymbol{e}_t + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c)$ as a modified error term, one that combines the original error with the error that would be induced by the rotational motion due to $\boldsymbol{\omega}_c$. The translational control input $\boldsymbol{v}_c = -\mathbf{L}_v^{+}(\lambda \boldsymbol{e}_t + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c)$ will drive this error to zero. The method known as 2.5-D visual servo [34.36] was the first to exploit such a partitioning in combining IBVS and PBVS. More precisely, in [34.36], $\boldsymbol{s}_t$ has been selected as the coordinates of an image point, and the logarithm of its depth, so that $\mathbf{L}_v$ is a triangular always invertible matrix. More precisely, we have $\boldsymbol{s}_t = (\boldsymbol{x}, \log Z)$, $\boldsymbol{s}_t^* = (\boldsymbol{x}^*, \log Z^*)$, $\boldsymbol{e}_t = (\boldsymbol{x} - \boldsymbol{x}^*, \log \rho_Z)$ where $\rho_Z = Z/Z^*$, and

$$\begin{aligned}\mathbf{L}_v &= \frac{1}{Z^* \rho_Z}\begin{pmatrix}-1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1\end{pmatrix}\mathbf{L}_{\boldsymbol{\omega}} \\ &= \begin{pmatrix}xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0\end{pmatrix}\,.\end{aligned}$$

Note that the ratio ρ_Z can be obtained directly from the partial pose estimation algorithm that will be described in Sect. 34.6.

If we come back to the usual global representation of visual servo control schemes, we have $\boldsymbol{e} = (\boldsymbol{e}_t, \theta \boldsymbol{u})$ and $\mathbf{L}_e$ given by

$$\mathbf{L}_e = \begin{pmatrix}\mathbf{L}_v & \mathbf{L}_{\boldsymbol{\omega}} \\ \mathbf{0} & \mathbf{L}_{\theta u}\end{pmatrix}\,,$$

from which we immediately obtain the control law (34.27) and (34.28) by applying (34.5).

The behavior obtained using this choice for $\boldsymbol{s}_t$ is shown in Fig. 34.12 and VIDEO 64. Here, the point that has been considered in $\boldsymbol{s}_t$ is the center of gravity $\boldsymbol{x}_g$ of the target. We note the image trajectory of that point, which is a straight line as expected, and the nice decreasing of the camera velocity components, which makes this scheme very similar to the first PBVS one.

As for stability, it is clear that this scheme is globally asymptotically stable in perfect conditions. Furthermore, thanks to the triangular form of the interaction matrix $\mathbf{L}_e$, it is possible to analyze the stability of this scheme in the presence of calibration errors using the partial pose-estimation algorithm that will be described in Sect. 34.6 [34.37]. Finally, the only unknown constant parameter involved in this scheme, that is Z^*, can be estimated online using adaptive techniques [34.38].

Other hybrid schemes can be designed. For instance, in [34.39], the third component of $\boldsymbol{s}_t$ is different and has been selected so that all the target points remain in the camera field of view as far as possible. Another example has been proposed in [34.40]. In that case, $\boldsymbol{s}$ is selected as $\boldsymbol{s} = ({}^{c^*}\boldsymbol{t}_c, \boldsymbol{x}_g, \theta u_z)$ which provides with a block-triangular interaction matrix of the form

$$\mathbf{L}_e = \begin{pmatrix}\mathbf{R} & \mathbf{0} \\ \mathbf{L}'_v & \mathbf{L}'_{\boldsymbol{\omega}}\end{pmatrix}\,,$$

where $\mathbf{L}'_v$ and $\mathbf{L}'_{\boldsymbol{\omega}}$ can easily be computed. This scheme is such that, under perfect conditions, the camera trajectory is a straight line (since ${}^{c^*}\boldsymbol{t}_c$ is a part of $\boldsymbol{s}$), and the image trajectory of the center of gravity of the object is also a straight line (since $\boldsymbol{x}_g$ is also a part of $\boldsymbol{s}$). The translational camera degrees of freedom are devoted to realize the 3-D straight line, while the rotational camera degrees of freedom are devoted to realize the 2-D straight line and also compensate the 2-D motion of $\boldsymbol{x}_g$ due to the translational motion. As can be seen in Fig. 34.13 and VIDEO 65, this scheme is particularly satisfactory in practice.

Finally, it is possible to combine 2-D and 3-D features in different ways. For instance, in [34.41], it has been proposed to use in $\boldsymbol{s}$ the 2-D homogeneous coordinates of a set of image points expressed in pixels multiplied by their corresponding depth:

Avidbots Ex1016
Page 888 of 2154

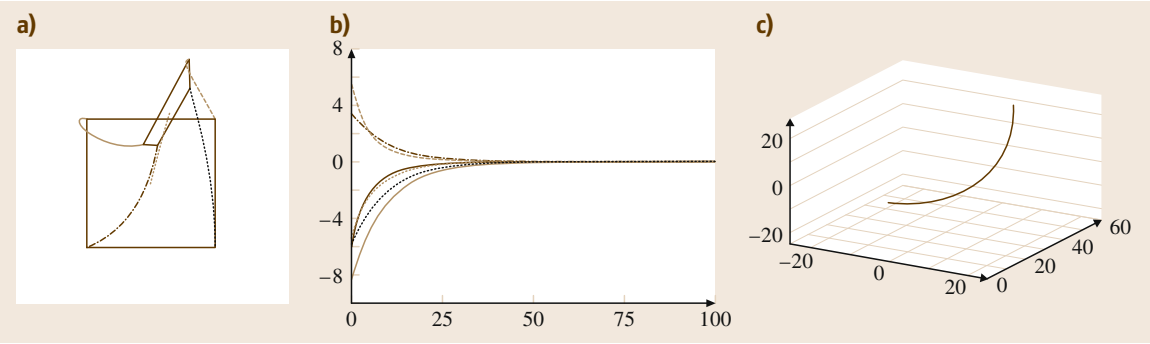


Fig. 34.12 2.5-D VS system behavior using $\boldsymbol{s} = (\mathbf{x}_g, \log(Z_g), \theta\boldsymbol{u})$ (refer to Fig. 34.2a-c for description)

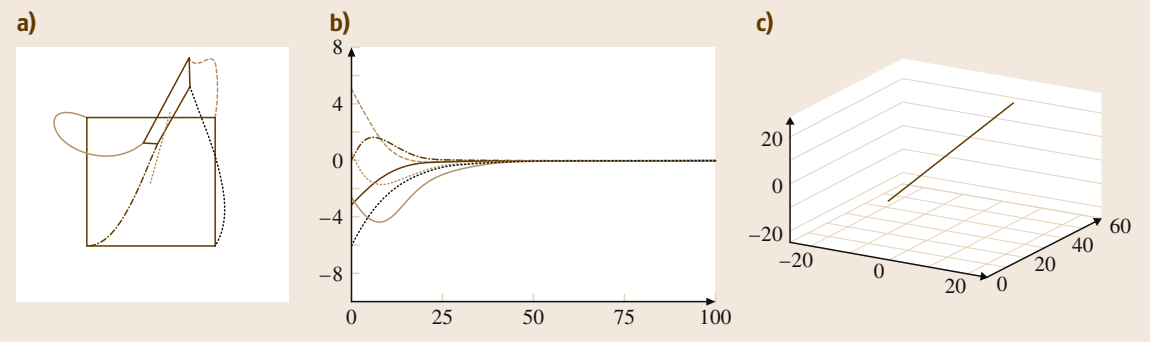


Fig. 34.13 2.5-D VS system behavior using $\boldsymbol{s} = ({}^{c^*}\mathbf{t}_c, \mathbf{x}_g, \theta u_z)$ (refer to Fig. 34.2a-c for description)

$\boldsymbol{s} = (u_1 Z_1, v_1 Z_1, Z_1, \cdots, u_n Z_n, v_n Z_n, Z_n)$. As for classical IBVS, we obtain in this case a set of redundant features, since at least three points have to be used to control the six camera degrees of freedom (here $k \geq 9$). However, it has been demonstrated in [34.42] that this selection of redundant features is free of attractive local minima.

34.4.2 Partitioned VS

The hybrid visual servo schemes described above have been designed to decouple the rotational motions from the translational ones by selecting adequate visual features defined in part in 2-D, and in part in 3-D (which is why they have been called 2.5-D visual servoing). This work has inspired some researchers to find features that exhibit similar decoupling properties but using only features expressed directly in the image. More precisely, the goal is to find six features such that each is related to only one degree of freedom (in which case the interaction matrix is a diagonal matrix). The Grail is to find a diagonal interaction matrix whose elements are constant, as near as possible to the identity matrix, leading to a pure, direct, and simple linear control problem.

The first work in this area partitioned the interaction matrix to isolate motion related to the optic axis [34.12]. Indeed, whatever the choice of $\boldsymbol{s}$, we have

$$\begin{aligned}\dot{\boldsymbol{s}} &= \mathbf{L}_{\boldsymbol{s}} \boldsymbol{v}_c \\ &= \mathbf{L}_{xy} \boldsymbol{v}_{xy} + \mathbf{L}_z \boldsymbol{v}_z \\ &= \dot{\boldsymbol{s}}_{xy} + \dot{\boldsymbol{s}}_z\end{aligned}$$

in which $\mathbf{L}_{xy}$ includes the first, second, fourth, and fifth columns of $\mathbf{L}_{\boldsymbol{s}}$, and $\mathbf{L}_z$ includes the third and sixth columns of $\mathbf{L}_{\boldsymbol{s}}$. Similarly, $\boldsymbol{v}_{xy} = (v_x, v_y, \omega_x, \omega_y)$ and $\boldsymbol{v}_z = (v_z, \omega_z)$. Here, $\dot{\boldsymbol{s}}_z = \mathbf{L}_z \boldsymbol{v}_z$ gives the component of $\dot{\boldsymbol{s}}$ due to the camera motion along and rotation about the optic axis, while $\dot{\boldsymbol{s}}_{xy} = \mathbf{L}_{xy} \boldsymbol{v}_{xy}$ gives the component of $\dot{\boldsymbol{s}}$ due to velocity along and rotation about the camera x and y axes.

Proceeding as above, by setting $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$ we obtain

$$-\lambda \boldsymbol{e} = \dot{\boldsymbol{e}} = \dot{\boldsymbol{s}} = \mathbf{L}_{xy} \boldsymbol{v}_{xy} + \mathbf{L}_z \boldsymbol{v}_z \,,$$

which leads to

$$\boldsymbol{v}_{xy} = -\mathbf{L}_{xy}^{+} [\lambda \boldsymbol{e}(t) + \mathbf{L}_z \boldsymbol{v}_z] \,.$$

As before, we can consider $[\lambda \boldsymbol{e}(t) + \mathbf{L}_z \boldsymbol{v}_z]$ as a modified error that incorporates the original error while taking into account the error that will be induced by $\boldsymbol{v}_z$.

Given this result, all that remains is to choose $\boldsymbol{s}$ and $\boldsymbol{v}_z$. As for basic IBVS, the coordinates of a collection of image points can be used in $\boldsymbol{s}$, while two new image features can be defined to determine $\boldsymbol{v}_z$:

- Define α, with $0 \leq \alpha < 2\pi$, as the angle between the horizontal axis of the image plane and the directed line segment joining two feature points.
  It is clear that α is closely related to the rotation around the optic axis.
- Define σ^2 to be the area of the polygon defined by these points. Similarly, σ^2 is closely related to the translation along the optic axis.

Using these features, $\boldsymbol{v}_z$ has been defined in [34.12] as

$$\begin{cases} v_z &= \lambda_{v_z} \ln \frac{\sigma^*}{\sigma} \,, \\ \omega_z &= \lambda_{\omega_z}(\alpha^* - \alpha) \,. \end{cases}$$

34.5 Performance Optimization and Planning

In some sense, partitioned methods represent an effort to optimize system performance by assigning distinct features and controllers to individual degrees of freedom. In this way, the designer performs a sort of offline optimization when allocating controllers to degrees of freedom. It is also possible to explicitly design controllers that optimize various system performance measures. We describe a few of these in this section.

34.5.1 Optimal Control and Redundancy Framework

An example of such an approach is given in [34.43] and [34.44], in which linear quadratic Gaussian (LQG) control design is used to choose gains that minimize a linear combination of state and control inputs. This approach explicitly balances the trade-off between tracking errors (since the controller attempts to drive $\boldsymbol{s} - \boldsymbol{s}^*$ to zero) and robot motion. A similar control approach is proposed in [34.45] where joint limit avoidance is considered simultaneously with the positioning task.

It is also possible to formulate optimality criteria that explicitly express the observability of robot motion in the image. For example, the singular value decomposition of the interaction matrix reveals which degrees of freedom are most apparent and can thus be easily controlled, while the condition number of the interaction matrix gives a kind of global measure of the visibility of motion. This concept has been called resolvability in [34.46] and motion perceptibility in [34.47]. By selecting features and designing controllers that maximize these measures, either along specific degrees of freedom or globally, the performance of the visual servo system can be improved.

The constraints considered to design the control scheme using the optimal control approach may be contradictory in some cases, leading the system to fail due to local minima in the objective function to be minimized. For example, it may happen that the motion produced to move away from a robot joint limit is exactly the opposite of the motion produced toward the desired pose, which results in a zero global motion. To avoid this potential problem, it is possible to use the gradient projection method, which is classical in robotics. Applying this method to visual servoing has been proposed in [34.3, 19]. The approach consists of projecting the secondary constraints $\boldsymbol{e}_s$ onto the null space of the vision-based task $\boldsymbol{e}$ so that they have no effect on the regulation of $\boldsymbol{e}$ to 0

$$\boldsymbol{e}_{\mathrm{g}} = \widehat{\mathbf{L}_e^+}\boldsymbol{e} + \mathbf{P}_e\,\boldsymbol{e}_s \,,$$

where $\boldsymbol{e}_{\mathrm{g}}$ is the new global task considered and

$$\mathbf{P}_e = \left(\mathbf{I}_6 - \widehat{\mathbf{L}_e^+}\widehat{\mathbf{L}_e}\right)$$

is such that

$$\widehat{\mathbf{L}_e}\mathbf{P}_e\boldsymbol{e}_s = 0 \,, \forall \boldsymbol{e}_s \,.$$

Avoiding the robot joint limits using this approach has been presented in [34.48]. However, when the vision-based task constrains all the camera degrees of freedom, the secondary constraints cannot be considered since, when $\widehat{\mathbf{L}_e}$ is of full rank 6, we have $\mathbf{P}_e\boldsymbol{e}_s = 0, \forall \boldsymbol{e}_s$. In this case, it is necessary to insert the constraints into a global objective function, such as navigation functions that are free of local minima [34.49, 50].

34.5.2 Switching Schemes

The partitioned methods described previously attempt to optimize performance by assigning individual controllers to specific degrees of freedom. Another way to use multiple controllers to optimize performance is to design switching schemes that select at each moment

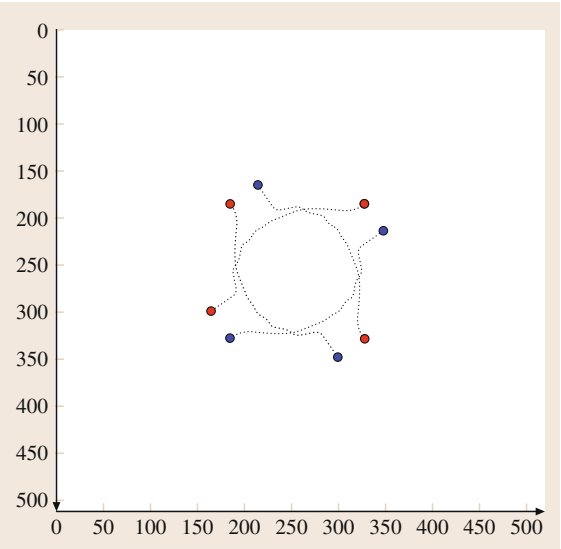


Fig. 34.14 Image feature trajectories for a rotation of 160° about the optical axis using a switched control scheme (initial point positions in *blue*, and desired points position in *red*)

in time which controller to use based on criteria to be optimized.

A simple switching controller can be designed using an IBVS and a PBVS controller as follows [34.51]. Let the system begin by using the IBVS controller. Consider the Lyapunov function for the PBVS controller given by $\mathcal{L}_P = \frac{1}{2}\|\boldsymbol{e}_P(t)\|^2$, with $\boldsymbol{e}_P(t) = ({}^c\boldsymbol{t}_o - {}^{c^*}\boldsymbol{t}_o, \theta\boldsymbol{u})$. If at any time the value of this Lyapunov function exceeds a threshold γ_P, the system switches to the PBVS controller. While using the PBVS controller, if at any time the value of the Lyapunov function $\mathcal{L}_I$ for the IBVS controller exceeds a threshold, $\mathcal{L}_I = \frac{1}{2}\|\boldsymbol{e}_I(t)\|^2 > \gamma_I$, the system switches to the IBVS controller. With this scheme, when the Lyapunov function for a particular controller exceeds a threshold, that controller is invoked, which in turn reduces the value of the corresponding Lyapunov function. If the switching thresholds are selected appropriately, the system is able to exploit the relative advantages of IBVS and PBVS, while avoiding their shortcomings.

An example of such a system is shown in Fig. 34.14 for the case of a rotation by 160° about the optical axis. Note that the system begins in IBVS mode and the features initially move on straight lines toward their goal positions in the image. However, as the camera retreats, the system switches to PBVS, which allows the camera to reach its desired position by combining a rotational motion around its optic axis and a forward translational motion, producing the circular trajectories observed in the image.

Other examples of temporal switching schemes can be found, such as the one developed in [34.52], to ensure the visibility of the target observed.

34.5.3 Feature Trajectory Planning

It is also possible to treat the optimization problem offline, during a planning stage, if we have sufficient knowledge of the system and world. In this case, several constraints can be taken into account simultaneously, such as obstacle avoidance [34.53], joint limit and occlusions avoidance, and ensuring the visibility of the target [34.54]. The feature trajectories $\boldsymbol{s}^*(t)$ that allow the camera to reach its desired pose while ensuring that the constraints are satisfied are determined using path planning techniques, such as the well-known potential field approach [34.54] or linear matrix inequality optimizations [34.55].

Coupling path planning with trajectory following also allows the robustness of the visual servo with respect to modeling errors to be significantly improved. Indeed, modeling errors may have large effects when the error $\boldsymbol{s}-\boldsymbol{s}^*$ is large, but have little effect when $\boldsymbol{s}-\boldsymbol{s}^*$ is small. Once the desired features trajectories $\boldsymbol{s}^*(t)$ such that $\boldsymbol{s}^*(0) = \boldsymbol{s}(0)$ have been designed during the planning stage, it is easy to adapt the control scheme to take into account the fact that $\boldsymbol{s}^*$ is varying, and to make the error $\boldsymbol{s}-\boldsymbol{s}^*$ remain small. More precisely, we now have

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{s}} - \dot{\boldsymbol{s}}^* = \mathbf{L}_{\boldsymbol{e}}\boldsymbol{v}_c - \dot{\boldsymbol{s}}^* ,$$

from which we deduce, by selecting as usual $\dot{\boldsymbol{e}} = -\lambda\boldsymbol{e}$ as the desired behavior,

$$\boldsymbol{v}_c = -\lambda\widehat{\mathbf{L}_{\boldsymbol{e}}^+}\boldsymbol{e} + \widehat{\mathbf{L}_{\boldsymbol{e}}^+}\dot{\boldsymbol{s}}^* .$$

The new second term of this control law anticipates the variation of $\boldsymbol{s}^*$, removing the tracking error it would produce. We will see in the 34.8 that the form of the control law is similar when tracking of a moving target is considered.

Avidbots Ex1016
Page 891 of 2154

34.6 Estimation of 3-D Parameters

All the control schemes described in the previous sections use 3-D parameters that are not directly available from the image measurements. As for IBVS, we recall that the range of the object with respect to the camera appears in the coefficients of the interaction matrix related to the translational degrees of freedom. Noticeable exceptions are the schemes based on a numerical estimation of $\mathbf{L}_e$ or of $\mathbf{L}_e^+$ (Sect. 34.2.8). Another exception is the IBVS scheme that uses the constant matrix $\widehat{\mathbf{L}_{e^*}^+}$ in the control scheme, in which only the depth for the desired pose is required, which is not so difficult to obtain in practice. As for PBVS and hybrid schemes that combine 2-D and 3-D data in $\boldsymbol{e}$, 3-D parameters appear both in the error $\boldsymbol{e}$ and in the interaction matrix. A correct estimation of the 3-D parameters involved is thus important for IBVS since they will have an effect on the camera motion during the task execution (they appear in the stability conditions (34.13) and (34.14)), while a correct estimation is crucial in PBVS and hybrid schemes since they will have also an effect on the accuracy of the pose reached after convergence.

If a calibrated stereo vision system is used, all 3-D parameters can be easily determined by triangulation, as mentioned in Sect. 34.2.5 and described in Chap. 32. Similarly, if a 3-D model of the object is known, all 3-D parameters can be computed from a pose-estimation algorithm. However, we recall that such an estimation can be quite unstable due to image noise (Sect. 34.3). As already said, IBVS does not require full pose estimation, simply the range of the object with respect to the camera. When image points are involved in $\boldsymbol{s}$, the range is expressed as the scalar depth Z or distance R of the corresponding world points, which appears in the interaction matrix (34.12), (34.16) and (34.17). This can be considered as a parameter estimation problem, that is, estimating the 3-D parameters from knowledge of the analytical form of the interaction matrix, and measurements of camera motion and visual feature position and velocity [34.56–58].

It is also possible to estimate 3-D parameters by using the epipolar geometry that relates the images of the same scene observed from different viewpoints. Indeed, in visual servoing, two images are generally available: the current one and the desired one. Given a set of matches between the image measurements in the current image and in the desired one, the fundamental matrix, or the essential matrix if the camera is calibrated, can be recovered [34.6], and then used in visual servoing [34.59]. Indeed, from the essential matrix, the rotation and the translation up to a scalar factor between the two views can be estimated. However, near the convergence of the visual servo, that is, when the current and desired images are similar, the epipolar geometry becomes degenerate and it is not possible to estimate accurately the partial pose between the two views. For this reason, using homography is generally preferred.

Let $\boldsymbol{x}_i$ and $\boldsymbol{x}_i^*$ denote the homogeneous image coordinates for a point in the current and desired images. Then $\boldsymbol{x}_i$ is related to $\boldsymbol{x}_i^*$ by

$$\boldsymbol{x}_i = \mathbf{H}_i \boldsymbol{x}_i^*$$

in which $\mathbf{H}_i$ is a homography matrix.

If all feature points lie on a 3-D plane, then there is a single homography matrix $\mathbf{H}$ such that $\boldsymbol{x}_i = \mathbf{H}\boldsymbol{x}_i^*$ for all i. This homography can be estimated using the position of four matched points in the desired and the current images. If all the features points do not belong to the same 3-D plane, then three points can be used to define such a plane and five supplementary points are needed to estimate $\mathbf{H}$ [34.60].

Once $\mathbf{H}$ is available, it can be decomposed as

$$\mathbf{H} = \mathbf{R} + \frac{\boldsymbol{t}}{d^*}\boldsymbol{n}^{*\mathrm{T}}, \tag{34.29}$$

in which $\mathbf{R}$ is the rotation matrix relating the orientation of the current and desired camera frames, $\boldsymbol{n}^*$ is the normal to the chosen 3-D plane expressed in the desired frame, d^* is the distance to the 3-D plane from the desired frame, and $\boldsymbol{t}$ is the translation between current and desired frames. From $\mathbf{H}$, it is thus possible to recover $\mathbf{R}$, $\boldsymbol{t}/d^*$, and $\boldsymbol{n}$. In fact, two solutions for these quantities exist [34.61], but it is quite easy to select the correct one using some knowledge about the desired pose. It is also possible to estimate the depth of any target point up to a common scale factor [34.54]. The unknown depth of each point that appears in classical IBVS can thus be expressed as a function of a single, constant parameter whatever the number of points. Similarly, the pose parameters required by PBVS can be recovered up to a scalar factor as for the translation term. The PBVS schemes described previously can thus be revisited using this approach, with the new error defined as the translation up to a scalar factor and the angle/axis parameterization of the rotation. This approach has also been used for the hybrid visual servoing schemes described in Sect. 34.4.1. In that case, using such homography estimation, it has been possible to analyze the stability of hybrid visual servoing schemes in the presence of calibration errors [34.36]. Finally, it is also possible to directly use the homography in the control scheme, avoiding thus its decomposition as a partial pose [34.62].

Avidbots Ex1016
Page 892 of 2154

34.7 Determining s^* and Matching Issues

All visual servo methods require knowledge of the desired feature values $\boldsymbol{s}^*$ which implicitly define constraints on the desired camera or robot pose with respect to the target. Three common approaches are employed. The first is when the task is directly specified as a desired value of some features to be reached. This is the case for instance when a target has to be centered in the image. This is also the case when the task is specified as a particular pose to reach and a PBVS is chosen. In that case however, the camera will reach its desired pose only if the camera is perfectly calibrated. Indeed, a coarse camera calibration will induce a biased estimation of the pose, which will make the final pose different from the desired one.

For IBVS and hybrid schemes, the second approach is to use knowledge of the object and camera projection models to compute $\boldsymbol{s}^*$ for the desired relative pose. Once again, the accuracy of the system directly depends on the camera calibration, since the camera intrinsic parameters are involved to compute $\boldsymbol{s}^*$.

Finally, the third approach is to simply record the feature values $\boldsymbol{s}^*$ when the camera or robot has the desired pose with respect to the target. This is usually done during an off-line teaching step. When it is possible in practice, this approach is very efficient since the positioning accuracy does not depend anymore on the camera calibration. This is still true for PBVS since, even if the pose estimated from the desired image is biased due to calibration errors, the same biased pose will be estimated once the robot will have converged so that the final image acquired by the camera will be the desired one.

So far we have not commented on the matching issues involved in visual servo methods. Two cases can be differentiated in function of the nature of the components of $\boldsymbol{s}$. When some components of $\boldsymbol{s}$ come from a pose estimation, it is necessary to match the measurements in the image (usually some image points) to the model of the object (usually some world points). Incorrect association will lead to an erroneous estimation of the pose. In all other cases, the calculation of the error vector $\boldsymbol{e}$ defined in (34.1) necessitates a matching between the measurements $\boldsymbol{m}(t)$ in the current image and $\boldsymbol{m}^*$ in the desired image. For instance, in all IBVS examples presented above, the camera observes four image points and we need to determine which of the four desired points to associate with each observed point so that the visual features $\boldsymbol{s}$ are correctly associated to their desired value $\boldsymbol{s}^*$. Incorrect association will lead to incorrect final camera pose and possibly a configuration which can not be achieved from any real camera pose. If we want to use the epipolar geometry or estimate a homography matrix, a similar matching process is necessary (Sect. 34.6).

This matching process is a classical computer vision problem. Note that it may be particularly difficult for the very first image, especially when the robot displacement to achieve is large, which generally implies large disparities between the initial and desired images. Once the association has been correctly performed for the very first image, the matching is greatly simplified since it transforms to a visual tracking problem where the results obtained for the previous image can be used as initialization for the current one.

34.8 Target Tracking

We now consider the case of a moving target and a constant desired value $\boldsymbol{s}^*$ for the features, the generalization to varying desired features $\boldsymbol{s}^*(t)$ being immediate. The time variation of the error is now given by

$$\dot{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{e}} \boldsymbol{v}_{\mathrm{c}} + \frac{\partial \boldsymbol{e}}{\partial t} , \tag{34.30}$$

where the term $\partial \boldsymbol{e}/\partial t$ expresses the time variation of $\boldsymbol{e}$ due to the generally unknown target motion. If the control law is still designed to try to ensure an exponential decoupled decrease of $\boldsymbol{e}$ (that is, once again $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$), we now obtain using (34.30)

$$\boldsymbol{v}_{\mathrm{c}} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} - \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} , \tag{34.31}$$

where $\widehat{\partial \boldsymbol{e}/\partial t}$ is an estimation or an approximation of $\partial \boldsymbol{e}/\partial t$. This term must be introduced into the control law to compensate for the target motion.

Closing the loop, that is, inserting (34.31) into (34.30), we obtain

$$\dot{\boldsymbol{e}} = -\lambda \mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} - \mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} + \frac{\partial \boldsymbol{e}}{\partial t} . \tag{34.32}$$

Even if $\mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} > 0$, the error will converge to zero only if the estimation of $\widehat{\partial \boldsymbol{e}/\partial t}$ is sufficiently accurate so that

$$\mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \frac{\partial \boldsymbol{e}}{\partial t} , \tag{34.33}$$

Avidbots Ex1016
Page 893 of 2154

otherwise tracking errors will be observed. Indeed, by just solving the scalar differential equation $\dot{e} = -\lambda e + b$, which is a simplification of (34.32), we obtain $e(t) = e(0)\exp(-\lambda t) + b/\lambda$, which converges towards b/λ. On one hand, setting a high gain λ will reduce the tracking error, but on the other hand, setting the gain too high can make the system unstable. It is thus necessary to make b as small as possible.

Of course, if the system is known to be such that $\partial \boldsymbol{e}/\partial t = 0$ (that is, the camera observes a motionless object, as described in Sect. 34.1), no tracking error will appear with the most simple estimation given by $\widehat{\partial \boldsymbol{e}/\partial t} = 0$. Otherwise, a classical method in automatic control to cancel tracking errors consists of compensating the target motion through an integral term in the control law. In this case, wehave

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \mu \sum_j \boldsymbol{e}(j) \,,$$

where μ is the integral gain that has to be tuned. This scheme allows the tracking errors to be canceled only if the target has a constant velocity. Other methods, based on feedforward control, estimate the term $\widehat{\partial \boldsymbol{e}/\partial t}$ directly through the image measurements and the camera velocity, when it is available. Indeed, from (34.30), we obtain

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \widehat{\dot{\boldsymbol{e}}} - \widehat{\mathbf{L}_e}\,\widehat{\boldsymbol{v}_c} \,,$$

where $\widehat{\dot{\boldsymbol{e}}}$ can, for instance, be obtained as $\widehat{\dot{\boldsymbol{e}}}(t) = [\boldsymbol{e}(t) - \boldsymbol{e}(t-\Delta t)]/\Delta t$, Δt being the duration of the control loop. A Kalman filter [34.63] or more-elaborate filtering methods [34.64] can then be used to improve the estimated values obtained. If some knowledge about the target velocity or the target trajectory is available, it can of course be used to smooth or predict the motion [34.65–67]. For instance, in [34.68], the periodic motion of the heart and breathing are compensated for an application of visual servoing in medical robotics. Finally, other methods have been developed to remove the perturbations induced by the target motion as fast as possible [34.43], using for instance predictive controllers [34.69].

34.9 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space

In the previous sections, we have considered the six components of the camera velocity as the input of the robot controller. As soon as the robot is not able to realize this motion, for instance, because it has fewer than six degrees of freedom, the control scheme must be expressed in the joint space. In this section, we describe how this can be done, and in the process develop a formulation for eye-to-hand systems.

In the joint space, the system equations for both the eye-to-hand and eye-in-hand configurations have the same form

$$\dot{\boldsymbol{s}} = \mathbf{J}_s\,\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{s}}{\partial t} \,. \tag{34.34}$$

Here, $\mathbf{J}_s \in \mathbb{R}^{k\times n}$ is the feature Jacobian matrix, which can be linked to the interaction matrix, and n is the number of robot joints.

For an eye-in-hand system (Fig. 34.15a), $\partial \boldsymbol{s}/\partial t$ is the time variation of $\boldsymbol{s}$ due to a potential object motion, and $\mathbf{J}_s$ is given by

$$\mathbf{J}_s = \mathbf{L}_s\,{}^{c}\boldsymbol{X}_N \mathbf{J}(\boldsymbol{q}) \,, \tag{34.35}$$

where:

- ${}^{c}\boldsymbol{X}_N$ is the spatial motion transform matrix (as defined in Chap. 02 and recalled in (34.15)) from the vision sensor frame to the end-effector frame. It is usually a constant matrix (as long as the vision sensor is rigidly attached to the end-effector). Thanks to the robustness of closed-loop control schemes, a coarse approximation of this transform matrix is sufficient in visual servoing. If needed, an accurate estimation is possible through classical hand–eye calibration methods [34.70].
- $\mathbf{J}(\boldsymbol{q})$ is the robot Jacobian expressed in the end-effector frame (as defined in Chap. 02)

For an eye-to-hand system (Fig. 34.15b), $\partial \boldsymbol{s}/\partial t$ is now the time variation of $\boldsymbol{s}$ due to a potential vision sensor motion and $\mathbf{J}_s$ can be expressed as

$$\mathbf{J}_s = -\mathbf{L}_s\,{}^{c}\boldsymbol{X}_N\,{}^{N}\mathbf{J}(\boldsymbol{q}) \,, \tag{34.36}$$

$$= -\mathbf{L}_s\,{}^{c}\boldsymbol{X}_0\,{}^{0}\mathbf{J}(\boldsymbol{q}) \,. \tag{34.37}$$

In (34.36), the classical robot Jacobian ${}^{N}\mathbf{J}(\boldsymbol{q})$ expressed in the end-effector frame is used but the spatial motion transform matrix ${}^{c}\boldsymbol{X}_N$ from the vision sensor frame to the end-effector frame changes all along the robot mo-

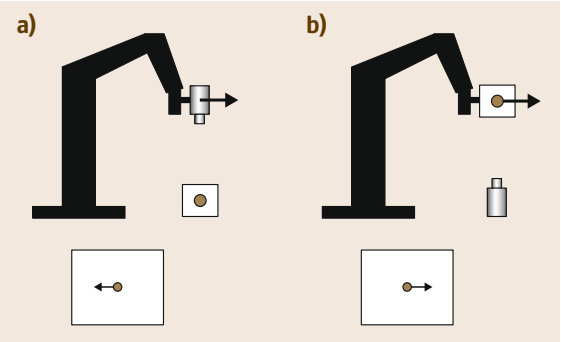


Fig.34.15a,b **(a)** Eye-in-hand system, **(b)** eye-to-hand system: system schematic (*top*) and opposite image motion produced by the same robot motion (*bottom*)

tion, and it has to be estimated at each iteration of the control scheme, usually using pose-estimation methods.

In (34.37), the robot Jacobian ${}^0\mathbf{J}(\boldsymbol{q})$ is expressed in the robot reference frame, and the spatial motion transform matrix ${}^c\boldsymbol{X}_0$ from the vision sensor frame to that reference frame is constant as long as the camera does not move. In this case, which is convenient in practice, a coarse approximation of ${}^c\boldsymbol{X}_0$ is usually sufficient.

Once the modeling step is finished, it is quite easy to follow the procedure that has been used above to design a control scheme expressed in the joint space, and to determine the sufficient condition to ensure the stability of the control scheme. We obtain, considering again $\boldsymbol{e} = \boldsymbol{s} - \boldsymbol{s}^*$, and an exponential decoupled decrease of $\boldsymbol{e}$

$$\dot{\boldsymbol{q}} = -\lambda \widehat{\mathbf{J}_e^+} \boldsymbol{e} - \widehat{\mathbf{J}_e^+} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} \,. \tag{34.38}$$

If $k = n$, considering as in Sect. 34.1 the Lyapunov function $\mathcal{L} = \frac{1}{2}\|\boldsymbol{e}(t)\|^2$, a sufficient condition to ensure the global asymptotic stability is given by

$$\mathbf{J}_e \widehat{\mathbf{J}_e^+} > 0 \,. \tag{34.39}$$

If $k > n$, we obtain similarly to Sect. 34.1

$$\widehat{\mathbf{J}_e^+} \mathbf{J}_e > 0 \tag{34.40}$$

to ensure the local asymptotic stability of the system. Note that the actual extrinsic camera parameters appear in $\mathbf{J}_e$ while the estimated ones are used in $\widehat{\mathbf{J}_e^+}$. It is thus possible to analyze the robustness of the control scheme with respect to the camera extrinsic parameters. It is also possible to estimate directly the numerical value of $\mathbf{J}_e$ or $\mathbf{J}_e^+$ using the methods described in Sect. 34.2.8.

Finally, to remove tracking errors, we have to ensure that

$$\mathbf{J}_e \widehat{\mathbf{J}_e^+} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \frac{\partial \boldsymbol{e}}{\partial t} \,.$$

Let us note that, even if the robot has six degrees of freedom, it is generally not equivalent to first compute $\boldsymbol{v}_c$ using (34.5) and then deduce $\dot{\boldsymbol{q}}$ using the robot inverse Jacobian, and to compute directly $\dot{\boldsymbol{q}}$ using (34.38). Indeed, it may occur that the robot Jacobian $\mathbf{J}(\boldsymbol{q})$ is singular while the feature Jacobian $\mathbf{J}_s$ is not (that may occur when $k < n$). Furthermore, the properties of the pseudo-inverse ensure that using (34.5), $\|\boldsymbol{v}_c\|$ is minimal while using (34.38), $\|\dot{\boldsymbol{q}}\|$ is minimal. As soon as

$$\mathbf{J}_e^+ \neq \mathbf{J}^+(\boldsymbol{q})^N \boldsymbol{X}_c \mathbf{L}_e^+ \,,$$

the control schemes will be different and will induce different robot trajectories. The choice of the state space is thus important.

34.10 Under Actuated Robots

Many useful robots are under actuated, that is, they cannot move instantaneously in all directions because of the number or configuration of their actuators (Chap. 52). For instance, a quadrotor flying robot is under actuated since it has only four actuators while the dimension of its configuration space is six. Many other useful robots are subject to non-holonomic constraints (Chap. 49), leading to similar motion inability. A car for instance has only two degrees of freedom (for velocity and steering) while the dimension of its configuration space is three (position and orientation in the ground plane).

A consequence of under actuation is that time varying manoeuvres may be required in order to achieve particular goal states. For example, if a quadrotor has to move forward, it must first change its attitude, pitching down, so that a component of its thrust vector is able to accelerate the vehicle forward. For a visual servo system this can be problematic since the initial attitude change will affect the value of the visual features even before the vehicle has moved. In fact the attitude change increases the error and this is ultimately destabilising.

A common and expedient solution [34.71] is to *de-rotate* the image, that is, to use information from

Avidbots Ex1016
Page 895 of 2154

a non-vision attitude sensor such as an inertial measurement unit (IMU) (Chap. 29) to correct the feature coordinates as if they had been viewed by a virtual camera whose optical axis has a constant orientation in space, typically, straight down. As discussed in Sect. 34.2.8, the spherical projection model is well suited for such image transformation, thanks to its invariance properties with respect to rotational motion.

For a non-holonomic vehicle, there are several solutions. In the non-general case where the vehicle is able to follow a smooth path from its initial to goal configuration, a controller based on visually estimated relative pose (range, heading angle and lateral offset) can be used (Chap. 49), following a PBVS strategy. Particular controllers based on the epipolar geometry or the trifocal tensor have also been designed [34.72, 73]. For the more general case, switching control laws can be used. If possible in practice, another common solution is to add a controlled DOF between the vehicle and the camera in order to bypass the non-holonomic constraint through redundancy. It is thus possible to control the full camera configuration, but not the robot one.

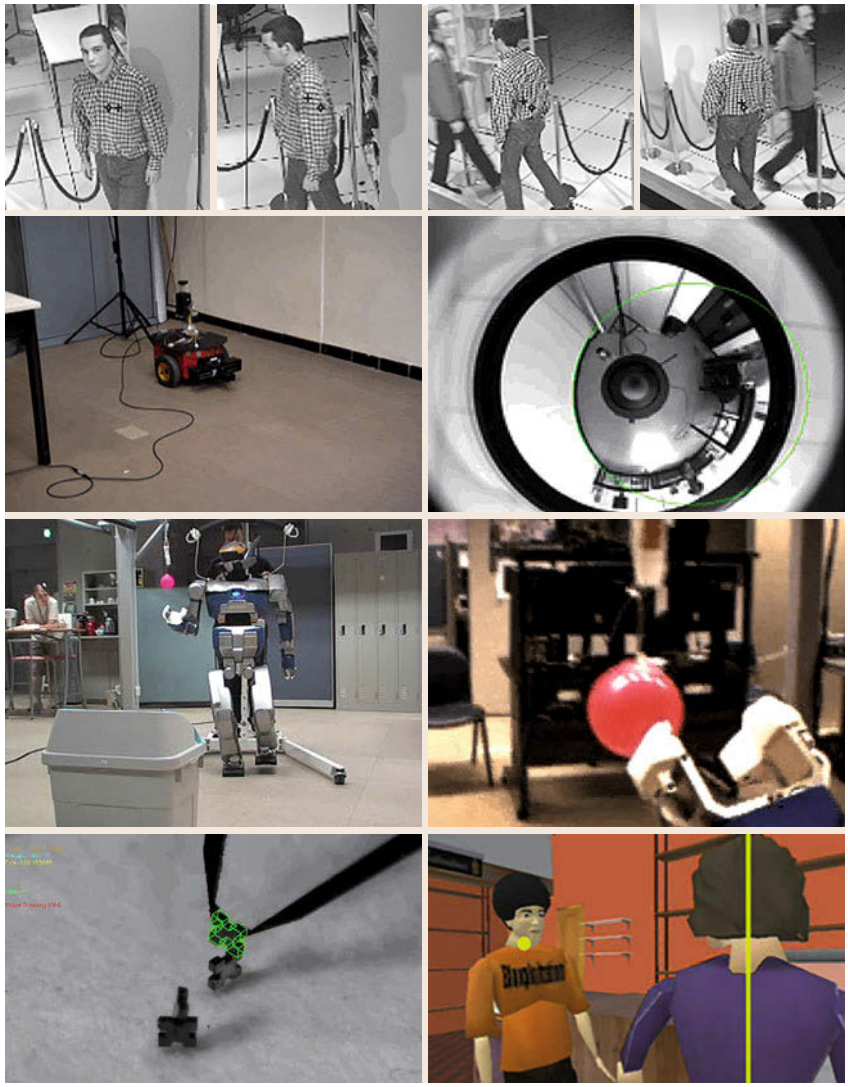


Fig. 34.16 Few applications of visual servoing: Gaze control for target tracking, navigation of a mobile robot to follow a wall using an omnidirectional vision sensor, grasping a ball with a humanoid robot, assembly of MEMS and film of a dialogue within the constraints of a script in animation

Avidbots Ex1016
Page 896 of 2154

34.11 Applications

Applications of visual servoing in robotics are numerous. It can be used as soon as a vision sensor is available and a task is assigned to a dynamic system to control its motion. A non exhaustive list of examples are (Fig. 34.16):

- The control of a pan-tilt-zoom camera for target tracking
- Grasping using a robot arm
- Locomotion and dextrous manipulation with a humanoid robot
- Micro or nano manipulation of microelectromechanical system (MEMS) or biological cells
- Accurate positioning of a parallel robot
- Pipe inspection by an underwater autonomous vehicle
- Autonomous navigation of a mobile robot in indoor or outdoor environment
- Aircraft landing
- Autonomous satellite rendezvous
- Biopsy using ultrasound probes or heart motion compensation in medical robotics
- Virtual cinematography in animation.

Software are available to help any user interested in developing visual servoing applications. We recommend the Robotics, Machine Vision Toolbox for Matlab [34.74] and the ViSP C++ library [34.11].

34.12 Conclusions

In this chapter we have only considered velocity controllers, which are convenient for most classical robot arms. However, the dynamics of the robot must of course be taken into account for high-speed tasks. Features related to the image motion [34.75], image intensity [34.76], or coming from other vision sensors (RGB-D sensors, ultrasonic probes [34.77], etc.) necessitate reconsideration of the modeling issues to select adequate visual features. Finally, fusing visual features with data coming from other sensors (force sensor, proximity sensors, etc.) at the level of the control scheme will allow new research topics to be addressed. The end of fruitful research in the field of visual servo is thus nowhere yet in sight.

Video-References

VIDEO 59 IBVS on a 6 DOF robot arm (1);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/59

VIDEO 60 IBVS on a 6 DOF robot arm (2);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/60

VIDEO 61 IBVS on a 6 DOF robot arm (3);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/61

VIDEO 62 PBVS on a 6 DOF robot arm (1);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/62

VIDEO 63 PBVS on a 6 DOF robot arm (2);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/63

VIDEO 64 2.5-D VS on a 6 DOF robot arm (1);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/64

VIDEO 65 2.5-D VS on a 6 DOF robot arm (2);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/65

References

34.1 L. Weiss, A. Sanderson, C. Neuman: Dynamic sensor-based control of robots with visual feedback, IEEE J. Robot. Autom. **3**, 404–417 (1987)

34.2 S. Hutchinson, G. Hager, P. Corke: A tutorial on visual servo control, IEEE Trans. Robot. Autom. **12**, 651–670 (1996)

34.3 B. Espiau, F. Chaumette, P. Rives: A new approach to visual servoing in robotics, IEEE Trans. Robot. Autom. **8**, 313–326 (1992)

34.4 J. Feddema, O. Mitchell: Vision-guided servoing with feature-based trajectory generation, IEEE Trans. Robot. Autom. **5**, 691–700 (1989)

Avidbots Ex1016
Page 897 of 2154

34.5 D. Forsyth, J. Ponce: *Computer Vision: A Modern Approach* (Prentice Hall, Upper Saddle River 2003)
34.6 Y. Ma, S. Soatto, J. Kosecka, S. Sastry: *An Invitation to 3-D Vision: From Images to Geometric Models* (Springer, New York 2003)
34.7 P. Corke: *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer, Berlin, Heidelberg 2011)
34.8 H. Michel, P. Rives: Singularities in the Determination of the Situation of a Robot Effector from the Perspective View of Three Points. Res. Rep. RR-1850 (INRIA 1993)
34.9 M. Fischler, R. Bolles: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Communications ACM **24**, 381–395 (1981)
34.10 E. Malis: Improving vision-based control using efficient second-order minimization techniques, IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 1843–1848
34.11 E. Marchand, F. Spindler, F. Chaumette: ViSP for visual servoing: A generic software platform with a wide class of robot control skills, IEEE Robot. Autom. Mag. **12**(4), 40–52 (2005), https://team.inria.fr/lagadic/visp/visp.html
34.12 P. Corke, S. Hutchinson: A new partitioned approach to image-based visual servo control, IEEE Trans. Robot. Autom. **17**, 507–515 (2001)
34.13 F. Chaumette: Potential problems of stability and convergence in image-based and position-based visual servoing, Lect. Note. Contr. Inform. Sci. **237**, 66–78 (1998)
34.14 E. Malis: Visual servoing invariant to changes in camera intrinsic parameters, IEEE Trans. Robot. Autom. **20**, 72–81 (2004)
34.15 A. Isidori: *Nonlinear Control Systems*, 3rd edn. (Springer, Berlin, Heidelberg 1995)
34.16 G. Hager, W. Chang, A. Morse: Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination, IEEE Control Syst. Mag. **15**, 30–39 (1995)
34.17 M. Iwatsuki, N. Okiyama: A new formulation of visual servoing based on cylindrical coordinate system, IEEE Trans. Robot. Autom. **21**, 266–273 (2005)
34.18 F. Chaumette, P. Rives, B. Espiau: Classification and realization of the different vision-based tasks, Robot. Autom. Syst. **7**, 199–228 (1993)
34.19 A. Castano, S. Hutchinson: Visual compliance: Task directed visual servo control, IEEE Trans. Robot. Autom. **10**, 334–342 (1994)
34.20 G. Hager: A modular system for robust positioning using feedback from stereo vision, IEEE Trans. Robot. Autom. **13**, 582–595 (1997)
34.21 F. Chaumette: Image moments: A general and useful set of features for visual servoing, IEEE Trans. Robot. Autom. **20**, 713–723 (2004)
34.22 O. Tahri, F. Chaumette: Point-based and region-based image moments for visual servoing of planar objects, IEEE Trans. Robot. **21**, 1116–1127 (2005)
34.23 C. Geyer, K. Daniilidis: Catadioptric projective geometry, Int. J. Comput. Vis. **45**(3), 223–243 (2001)
34.24 T. Hamel, R. Mahony: Visual servoing of an under-actuated dynamic rigid-body system: An image-based approach, IEEE Trans Robot. **18**(2), 187–198 (2002)
34.25 I. Suh: Visual servoing of robot manipulators by fuzzy membership function based neural networks. In: *Visual Servoing*, Robotics and Automated Systems, Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 285–315
34.26 G. Wells, C. Venaille, C. Torras: Vision-based robot positioning using neural networks, Image Vis. Comput. **14**, 75–732 (1996)
34.27 J.T. Lapresté, F. Jurie, F. Chaumette: An efficient method to compute the inverse jacobian matrix in visual servoing, IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 727–732
34.28 K. Hosada, M. Asada: Versatile visual servoing without knowledge of true jacobian, IEEE/RSJ Int. Conf. Intell. Robots Syst., München (1994) pp. 186–193
34.29 M. Jägersand, O. Fuentes, R. Nelson: Experimental evaluation of uncalibrated visual servoing for precision manipulation, IEEE Int. Conf. Robot. Autom., Albuquerque (1997) pp. 2874–2880
34.30 J. Piepmeier, G.M. Murray, H. Lipkin: Uncalibrated dynamic visual servoing, IEEE Trans. Robot. Autom. **20**, 143–147 (2004)
34.31 K. Deguchi: Direct interpretation of dynamic images and camera motion for visual servoing without image feature correspondence, J. Robot. Mechatron. **9**(2), 104–110 (1997)
34.32 W. Wilson, C. Hulls, G. Bell: Relative end-effector control using cartesian position based visual servoing, IEEE Trans. Robot. Autom. **12**, 684–696 (1996)
34.33 B. Thuilot, P. Martinet, L. Cordesses, J. Gallice: Position based visual servoing: Keeping the object in the field of vision, IEEE Int. Conf. Robot. Autom., Washington (2002) pp. 1624–1629
34.34 D. Dementhon, L. Davis: Model-based object pose in 25 lines of code, Int. J. Comput. Vis. **15**, 123–141 (1995)
34.35 D. Lowe: Three-dimensional object recognition from single two-dimensional images, Artif. Intell. **31**(3), 355–395 (1987)
34.36 E. Malis, F. Chaumette, S. Boudet: 2-1/2 D visual servoing, IEEE Trans. Robot. Autom. **15**, 238–250 (1999)
34.37 E. Malis, F. Chaumette: Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods, IEEE Trans. Robot. Autom. **18**, 176–186 (2002)
34.38 J. Chen, D. Dawson, W. Dixon, A. Behal: Adaptive homography-based visual servo tracking for fixed camera-in-hand configurations, IEEE Trans. Control Syst. Technol. **13**, 814–825 (2005)
34.39 G. Morel, T. Leibezeit, J. Szewczyk, S. Boudet, J. Pot: Explicit incorporation of 2-D constraints in vision-based control of robot manipulators, Lect. Note. Contr. Inform. Sci. **250**, 99–108 (2000)

Avidbots Ex1016
Page 898 of 2154

34.40 F. Chaumette, E. Malis: 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings, IEEE Int. Conf. Robot. Autom., San Fransisco (2000) pp. 630–635

34.41 E. Cervera, A.D. Pobil, F. Berry, P. Martinet: Improving image-based visual servoing with three-dimensional features, Int. J. Robot. Res. **22**, 821–840 (2004)

34.42 F. Schramm, G. Morel, A. Micaelli, A. Lottin: Extended 2-D visual servoing, IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 267–273

34.43 N. Papanikolopoulos, P. Khosla, T. Kanade: Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control, IEEE Trans. Robot. Autom. **9**, 14–35 (1993)

34.44 K. Hashimoto, H. Kimura: LQ optimal and nonlinear approaches to visual servoing, Robot. Autom. Syst. **7**, 165–198 (1993)

34.45 B. Nelson, P. Khosla: Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance, Int. J. Robot. Res. **14**, 225–269 (1995)

34.46 B. Nelson, P. Khosla: Force and vision resolvability for assimilating disparate sensory feedback, IEEE Trans. Robot. Autom. **12**, 714–731 (1996)

34.47 R. Sharma, S. Hutchinson: Motion perceptibility and its application to active vision-based servo control, IEEE Trans. Robot. Autom. **13**, 607–617 (1997)

34.48 E. Marchand, F. Chaumette, A. Rizzo: Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing, IEEE/RSJ Int. Conf. Intell. Robots Syst., Osaka (1996) pp. 1083–1090

34.49 E. Marchand, G. Hager: Dynamic sensor planning in visual servoing, IEEE Int. Conf. Robot. Autom., Leuven (1998) pp. 1988–1993

34.50 N. Cowan, J. Weingarten, D. Koditschek: Visual servoing via navigation functions, IEEE Trans. Robot. Autom. **18**, 521–533 (2002)

34.51 N. Gans, S. Hutchinson: An asymptotically stable switched system visual controller for eye in hand robots, IEEE/RSJ Int. Conf. Intell. Robots Syst., Las Vegas (2003) pp. 735–742

34.52 G. Chesi, K. Hashimoto, D. Prattichizio, A. Vicino: Keeping features in the field of view in eye-in-hand visual servoing: a switching approach, IEEE Trans. Robot. Autom. **20**, 908–913 (2004)

34.53 K. Hosoda, K. Sakamato, M. Asada: Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3-D reconstruction, IEEE/RSJ Int. Conf. Intell. Robots Syst. 3, Pittsburgh (1995) pp. 29–34

34.54 Y. Mezouar, F. Chaumette: Path planning for robust image-based control, IEEE Trans. Robot. Autom. **18**, 534–549 (2002)

34.55 G. Chesi: Visual servoing path-planning via homogeneous forms and LMI optimizations, IEEE Trans. Robot. **25**(2), 281–291 (2009)

34.56 L. Matthies, T. Kanade, R. Szeliski: Kalman filter-based algorithms for estimating depth from image sequences, Int. J. Comput. Vis. **3**(3), 209–238 (1989)

34.57 C.E. Smith, N. Papanikolopoulos: Computation of shape through controlled active exploration, IEEE Int. Conf. Robot. Autom., San Diego (1994) pp. 2516–2521

34.58 A. De Luca, G. Oriolo, P. Robuffo Giordano: Feature depth observation for image-based visual servoing: Theory and experiments, Int. J. Robot. Res. **27**(10), 1093–1116 (2008)

34.59 R. Basri, E. Rivlin, I. Shimshoni: Visual homing: Surfing on the epipoles, Int. J. Comput. Vis. **33**, 117–137 (1999)

34.60 E. Malis, F. Chaumette, S. Boudet: 2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement, Int. J. Comput. Vis. **37**, 79–97 (2000)

34.61 O. Faugeras: *Three-Dimensional Computer Vision: A Geometric Viewpoint* (MIT Press, Cambridge 1993)

34.62 G. Silveira, E. Malis: Direct visual servoing: Vision-based estimation and control using only nonmetric information, IEEE Trans. Robot. **28**(4), 974–980 (2012)

34.63 P. Corke, M. Goods: Controller design for high performance visual servoing, 12th World Congr. IFAC'93, Sydney (1993) pp. 395–398

34.64 F. Bensalah, F. Chaumette: Compensation of abrupt motion changes in target tracking by visual servoing, IEEE/RSJ Int. Conf. Intell. Robots Syst., Pittsburgh (1995) pp. 181–187

34.65 P. Allen, B. Yoshimi, A. Timcenko, P. Michelman: Automated tracking and grasping of a moving object with a robotic hand-eye system, IEEE Trans. Robot. Autom. **9**, 152–165 (1993)

34.66 K. Hashimoto, H. Kimura: Visual servoing with non linear observer, IEEE Int. Conf. Robot. Autom., Nagoya (1995) pp. 484–489

34.67 A. Rizzi, D. Koditschek: An active visual estimator for dexterous manipulation, IEEE Trans. Robot. Autom. **12**, 697–713 (1996)

34.68 R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M.A. Sanchez, J. Marescaux: Active filtering of physiological motion in robotized surgery using predictive control, IEEE Trans. Robot. **21**, 67–79 (2005)

34.69 J. Gangloff, M. de Mathelin: Visual servoing of a 6-DOF manipulator for unknown 3-D profile following, IEEE Trans. Robot. Autom. **18**, 511–520 (2002)

34.70 R. Tsai, R. Lenz: A new technique for fully autonomous efficient 3-D robotics hand-eye calibration, IEEE Trans. Robot. Autom. **5**, 345–358 (1989)

34.71 N. Guenard, T. Hamel, R. Mahony: A practical visual servo control for an unmanned aerial vehicle, IEEE Trans. Robot. **24**(2), 331–340 (2008)

34.72 G.L. Mariottini, G. Oriolo, D. Prattichizo: Image-based visual servoing for nonholonomic mobile robots using epipolar geometry, IEEE Trans. Robot. **23**(1), 87–100 (2007)

34.73 G. Lopez-Nicolas, J.J. Guerrero, C. Sagues: Visual control through the trifocal tensor for nonholonomic robots, Robot. Auton. Syst. **58**(2), 216–226 (2010)

Avidbots Ex1016
Page 899 of 2154

34.74 P. Corke: *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, Springer Tracts in Advanced Robotics, Vol. 73 (Springer, Berlin, Heidelberg 2011)
34.75 A. Crétual, F. Chaumette: Visual servoing based on image motion, Int. J. Robot. Res. **20**(11), 857–877 (2001)
34.76 C. Collewet, E. Marchand: Photometric visual servoing, IEEE Trans. Robot. **27**(4), 828–834 (2011)
34.77 R. Mebarki, A. Krupa, F. Chaumette: 2D ultrasound probe complete guidance by visual servoing using image moments, IEEE Trans. Robot. **26**(2), 296–306 (2010)

Avidbots Ex1016
Page 900 of 2154

Multimedia Contents

35. Multisensor Data Fusion

Hugh Durrant-Whyte, Thomas C. Henderson

Multisensor data fusion is the process of combining observations from a number of different sensors to provide a robust and complete description of an environment or process of interest. Data fusion finds wide application in many areas of robotics such as object recognition, environment mapping, and localization.

This chapter has three parts: methods, architectures, and applications. Most current data fusion methods employ probabilistic descriptions of observations and processes and use Bayes' rule to combine this information. This chapter surveys the main probabilistic modeling and fusion techniques including grid-based models, Kalman filtering, and sequential Monte Carlo techniques. This chapter also briefly reviews a number of nonprobabilistic data fusion methods. Data fusion systems are often complex combinations of sensor devices, processing, and fusion algorithms. This chapter provides an overview of key principles in data fusion architectures from both a hardware and algorithmic viewpoint. The applications of data fusion are pervasive in robotics and underly the core problem of sensing, estimation, and perception. We highlight two example applications that bring out these features. The first describes a navigation or self-tracking application for an autonomous vehicle. The second describes an application in mapping and environment modeling.

35.1 **Multisensor Data Fusion Methods** 867
35.1.1 Bayes' Rule 868
35.1.2 Probabilistic Grids 869
35.1.3 The Kalman Filter 872
35.1.4 Sequential Monte Carlo Methods 876
35.1.5 Alternatives to Probability 878
35.2 **Multisensor Fusion Architectures** 880
35.2.1 Architectural Taxonomy 880
35.2.2 Centralized, Local Interaction, and Hierarchical 882
35.2.3 Decentralized, Global Interaction, and Heterarchical 882
35.2.4 Decentralized, Local Interaction, and Hierarchical 883
35.2.5 Decentralized, Local Interaction, and Heterarchical 884
35.3 **Applications** 885
35.3.1 Dynamic System Control 885
35.3.2 ANSER II: Decentralized Data Fusion 888
35.3.3 Recent Developments 889
35.4 **Conclusions** 889
Video-References 889
References 890

The essential algorithmic tools of data fusion are reasonably well established. However, the development and use of these tools in realistic robotics applications is still developing.

35.1 Multisensor Data Fusion Methods

The most widely used data fusion methods employed in robotics originate in the fields of statistics, estimation, and control. However, the application of these methods in robotics has a number of unique features and challenges. In particular, autonomy is most often the goal and so results must be presented and interpreted in a form from which autonomous decisions can be made, for recognition or navigation, for example.

In this section we review the main data fusion methods employed in robotics. These are very often based on probabilistic methods, which are indeed now considered the standard approach to data fusion in all robotics applications [35.1]. Probabilistic data fusion methods are generally based on Bayes' rule for combining prior and observation information. Practically, this may be implemented in a number of ways: through the use of the Kalman and extended Kalman filters, through sequential Monte Carlo methods, or through the use of functional density estimates. Each of these is reviewed. There are a number of alternatives to probabilistic methods. These include the theory of evidence and interval methods. Such alternative techniques are not as widely used as they once were, however they have some special features that can be advantageous in specific problems. These, too, are briefly reviewed.

35.1.1 Bayes' Rule

Bayes' rule lies at the heart of most data fusion methods. In general, Bayes' rule provides a means to make inferences about an object or environment of interest described by a state $\boldsymbol{x}$, given an observation z.

Bayesian Inference

Bayes' rule requires that the relationship between $\boldsymbol{x}$ and z be encoded as a joint probability or joint probability distribution $P(\boldsymbol{x}, z)$ for discrete and continuous variables, respectively. The chain rule of conditional probabilities can be used to expand a joint probability in two ways

$$P(\boldsymbol{x}, z) = P(\boldsymbol{x} \mid z)P(z) = P(z \mid \boldsymbol{x})P(\boldsymbol{x}). \tag{35.1}$$

Rearranging in terms of one of the conditionals, Bayes' rule is obtained

$$P(\boldsymbol{x} \mid z) = \frac{P(z \mid \boldsymbol{x})P(\boldsymbol{x})}{P(z)}. \tag{35.2}$$

The value of this result lies in the interpretation of the probabilities $P(\boldsymbol{x} \mid z)$, $P(z \mid \boldsymbol{x})$, and $P(\boldsymbol{x})$. Suppose it is necessary to determine the various likelihoods of different values of an unknown state $\boldsymbol{x}$. There may be prior beliefs about what values of $\boldsymbol{x}$ might be expected, encoded in the form of relative likelihoods in the *prior probability* $P(\boldsymbol{x})$. To obtain more information about the state $\boldsymbol{x}$ an observation z is made. These observations are modeled in the form of a conditional probability $P(z \mid \boldsymbol{x})$ that describes, for each fixed state $\boldsymbol{x}$, the probability that the observation z will be made, i. e., the probability of z given $\boldsymbol{x}$. The new likelihoods associated with the state $\boldsymbol{x}$ are computed from the product of the original prior information and the information gained by observation. This is encoded in the *posterior probability* $P(\boldsymbol{x} \mid z)$, which describes the likelihoods associated with $\boldsymbol{x}$ given the observation z. In this fusion process, the marginal probability $P(z)$ simply serves to normalize the posterior and is not generally computed. The marginal $P(z)$ plays an important role in model validation or data association as it provides a measure of how well the observation is predicted by the prior, because $P(z) = \int P(z \mid \boldsymbol{x})P(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$. The value of Bayes' rule is that it provides a principled means of combining observed information with prior beliefs about the state of the world.

Sensor Models and Multisensor Bayesian Inference

The conditional probability $P(z \mid \boldsymbol{x})$ serves the role of a *sensor model* and can be thought of in two ways. First, in building a sensor model, the probability is constructed by fixing the value of $\boldsymbol{x} = x$ and then asking what probability density $P(z \mid \boldsymbol{x} = x)$ on z results. Conversely, when this sensor model is used and observations are made, $z = z$ is fixed and a *likelihood function* $P(z = z \mid \boldsymbol{x})$ on $\boldsymbol{x}$ is inferred. The likelihood function, while not strictly a probability density, models the relative likelihood that different values of $\boldsymbol{x}$ gave rise to the observed value of z. The product of this likelihood with the prior, both defined on $\boldsymbol{x}$, gives the posterior or observation update $P(\boldsymbol{x} \mid z)$. In a practical implementation of (35.2), $P(z \mid \boldsymbol{x})$ is constructed as a function of both variables (or a matrix in discrete form). For each fixed value of $\boldsymbol{x}$, a probability density on z is defined. Therefore, as $\boldsymbol{x}$ varies, a family of likelihoods on z is created.

The multisensor form of Bayes' rule requires conditional independence

$$\begin{aligned} P(z_1, \cdots, z_n \mid \boldsymbol{x}) &= P(z_1 \mid \boldsymbol{x}) \cdots P(z_n \mid \boldsymbol{x}) \\ &= \prod_{i=1}^{n} P(z_i \mid \boldsymbol{x})\,, \end{aligned} \tag{35.3}$$

so that

$$P(\boldsymbol{x} \mid \boldsymbol{Z}^n) = CP(\boldsymbol{x}) \prod_{i=1}^{n} P(z_i \mid \boldsymbol{x})\,, \tag{35.4}$$

where C is a normalizing constant. Equation 35.4 is known as the *independent likelihood pool* [35.2]. This states that the posterior probability on $\boldsymbol{x}$ given all observations $\boldsymbol{Z}^n$, is simply proportional to the product of prior probability and individual likelihoods from each information source.

Avidbots Ex1016
Page 902 of 2154

The recursive form of Bayes' rule is

$$P(\boldsymbol{x} \mid \boldsymbol{Z}^k) = \frac{P(z_k \mid \boldsymbol{x}) P(\boldsymbol{x} \mid \boldsymbol{Z}^{k-1})}{P\left(z_k \mid \boldsymbol{Z}^{k-1}\right)} . \tag{35.5}$$

The advantage of (35.5) is that we need to compute and store only the posterior density $P(\boldsymbol{x} \mid \boldsymbol{Z}^{k-1})$, which contains a complete summary of all past information. When the next piece of information $P(z_k \mid \boldsymbol{x})$ arrives, the previous posterior takes on the role of the current prior and the product of the two becomes, when normalized, the new posterior.

Bayesian Filtering

Filtering is concerned with the sequential process of maintaining a probabilistic model for a state which evolves over time and which is periodically observed by a sensor. Filtering forms the basis for many problems in tracking and navigation. The general filtering problem can be formulated in Bayesian form. This is significant because it provides a common representation for a range of discrete and continuous data fusion problems without recourse to specific target or observation models.

Define $\boldsymbol{x}_t$ as the value of a state of interest at time t. This may, for example, describe a feature to be tracked, the state of a process being monitored, or the location of a platform for which navigation data is required. For convenience, and without loss of generality, time is defined at discrete (asynchronous) times $t_k \triangleq k$. At a time instant k, the following quantities are defined:

- $\boldsymbol{x}_k$: The state vector to be estimated at time k.
- $\boldsymbol{u}_k$: A control vector, assumed known, and applied at time $k-1$ to drive the state from $\boldsymbol{x}_{k-1}$ to $\boldsymbol{x}_k$ at time k.
- z_k: An observation taken of the state $\boldsymbol{x}_k$ at time k.

In addition, the following sets are also defined:

- The history of states: $\boldsymbol{X}^k = \{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_k\} = \{\boldsymbol{X}^{k-1}, \boldsymbol{x}_k\}$.
- The history of control inputs: $\boldsymbol{U}^k = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k\} = \{\boldsymbol{U}^{k-1}, \boldsymbol{u}_k\}$.
- The history of state observations: $\boldsymbol{Z}^k = \{z_1, z_2, \cdots, z_k\} = \{\boldsymbol{Z}^{k-1}, z_k\}$.

In probabilistic form, the general data fusion problem is to find the posterior density

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \tag{35.6}$$

for all times k given the recorded observations and control inputs up to and including time k together (possibly) with knowledge of the initial state $\boldsymbol{x}_0$. Bayes' rule can be used to write (35.6) in terms of a sensor model $P(z_k \mid \boldsymbol{x}_k)$ and a predicted probability density $P(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0)$ based on observations up to time $k-1$ as

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k, \boldsymbol{U}^k, \boldsymbol{x}_0\right) = \frac{P(z_k \mid \boldsymbol{x}_k) P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right)}{P\left(z_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k\right)} . \tag{35.7}$$

The denominator in (35.7) is independent of the state and following (35.4) can be set to some normalizing constant C. The sensor model makes use of the conditional independence assumption from (35.3).

The total probability theorem can be used to rewrite the second term in the numerator of (35.7) in terms of the state transition model and the joint posterior from time step $k-1$ as

$$\begin{aligned} & P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \\ &= \int P\left(\boldsymbol{x}_k, \boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \mathrm{d}\boldsymbol{x}_{k-1} \\ &= \int P\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \\ &\quad \times P\left(\boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \mathrm{d}\boldsymbol{x}_{k-1} \\ &= \int P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_k) \\ &\quad \times P\left(\boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^{k-1}, \boldsymbol{x}_0\right) \mathrm{d}\boldsymbol{x}_{k-1} , \end{aligned} \tag{35.8}$$

where the last equality implies that the future state depends only on the current state and the control exerted at this time. The state transition model is described in terms of a probability distribution in the form $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_k)$. That is, the state transition may reasonably be assumed to be a Markov process in which the next state $\boldsymbol{x}_k$ depends only on the immediately proceeding state $\boldsymbol{x}_{k-1}$ and the applied control $\boldsymbol{u}_k$, and is independent of both the observations and preceding states.

Equations (35.7) and (35.8) define a recursive solution to (35.6). Equation (35.8) is the *time update* or prediction step for the full Bayes data fusion algorithm. A graphical description of this equation is shown in Fig. 35.1. Equation (35.7) is the *observation update* step for the full Bayes data fusion algorithm. A graphical description of this equation is shown in Fig. 35.2. The Kalman filter, grid-based methods, and sequential Monte Carlo methods, to be described, are specific implementations of these general equations.

35.1.2 Probabilistic Grids

Probabilistic grids are conceptually the simplest way of implementing Bayesian data fusion methods. They can

Avidbots Ex1016
Page 903 of 2154

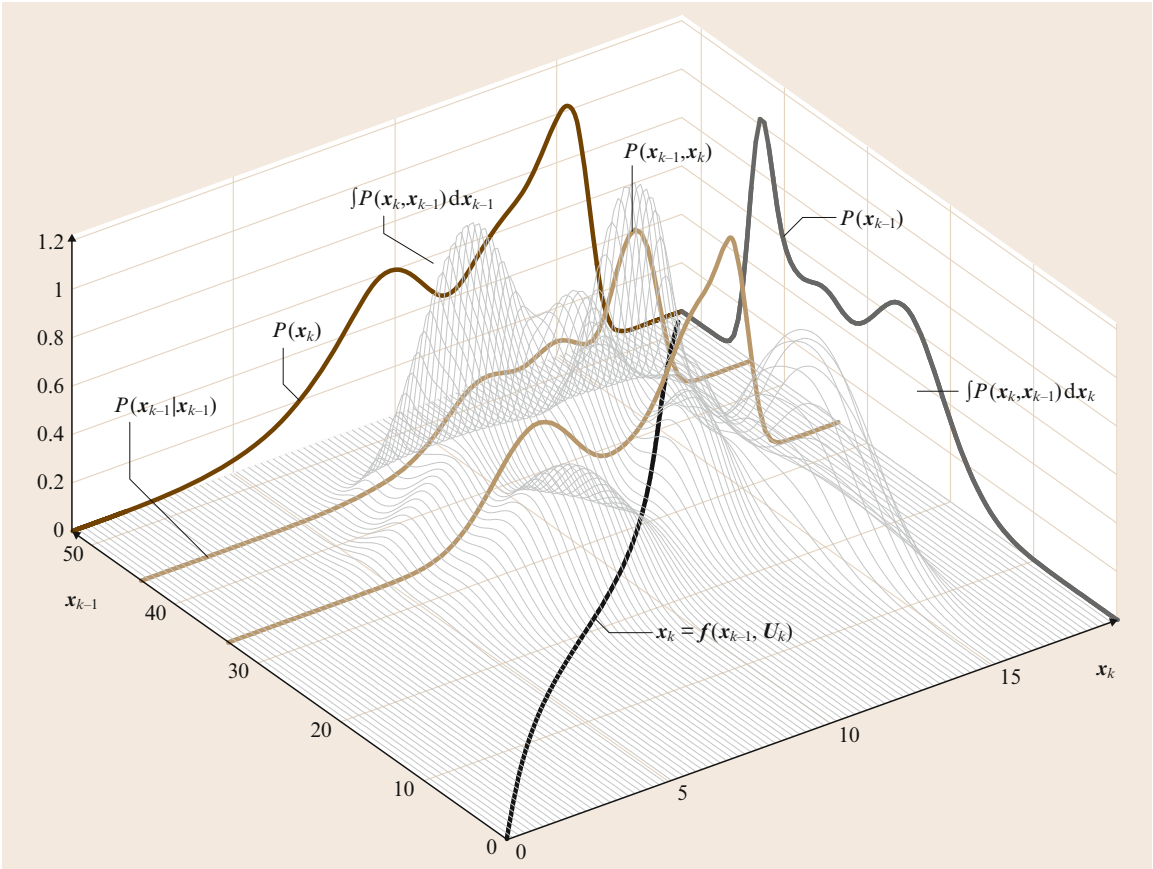


Fig. 35.1 Time update step for the full Bayes filter. At a time $k-1$, knowledge of the state $\boldsymbol{x}_{k-1}$ is summarized in a probability distribution $P(\boldsymbol{x}_{k-1})$. A vehicle model, in the form of a conditional probability density $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$, then describes the stochastic transition of the vehicle from a state $\boldsymbol{x}_{k-1}$ at a time $k-1$ to a state $\boldsymbol{x}_k$ at a time k. Functionally, this state transition may be related to an underlying kinematic state model in the form $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k)$. The figure shows two typical conditional probability distributions $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ on the state $\boldsymbol{x}_k$ given fixed values of $\boldsymbol{x}_{k-1}$. The product of this conditional distribution with the marginal distribution $P(\boldsymbol{x}_{k-1})$, describing the prior likelihood of values of $\boldsymbol{x}_k$, gives the the joint distribution $P(\boldsymbol{x}_k, \boldsymbol{x}_{k-1})$ shown as the surface in the figure. The total marginal density $P(\boldsymbol{x}_k)$ describes knowledge of $\boldsymbol{x}_k$ after state transition has occurred. The marginal density $P(\boldsymbol{x}_k)$ is obtained by integrating (projecting) the joint distribution $P(\boldsymbol{x}_k, \boldsymbol{x}_{k-1})$ over all $\boldsymbol{x}_{k-1}$. Equivalently, using the total probability theorem, the marginal density can be obtained by integrating (summing) all conditional densities $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ weighted by the prior probability $P(\boldsymbol{x}_{k-1})$ of each $\boldsymbol{x}_{k-1}$. The process can equally be run in reverse (a retroverse motion model) to obtain $P(\boldsymbol{x}_{k-1})$ from $P(\boldsymbol{x}_k)$ given a model $P(\boldsymbol{x}_{k-1} \mid \boldsymbol{x}_k)$

be applied both to problems in mapping [35.3, 4] and tracking [35.5].

In mapping applications, the environment of interest is divided into a grid of equally sized spatial cells. Each cell is indexed and labeled with a property, thus the state $\boldsymbol{x}_{ij}$ may describe a two-dimensional world indexed by ij and having the property x. Interest is focused on maintaining a probability distribution on possible state values $P(\boldsymbol{x}_{ij})$ *at each grid cell*. Typically, in navigation and mapping problems, the property of interest has only two values O and E, *occupied* and *empty*, respectively, and it is then usual to assume that $P(\boldsymbol{x}_{ij} = O) = 1 - P(\boldsymbol{x}_{ij} = E)$. However, there is no particular constraint on the property encoded by the state $\boldsymbol{x}_{ij}$ which could have many values (green, red, blue, for example) and indeed be continuous (the temperature at a cell, for example).

Once the state has been defined, Bayesian methods require that a sensor model or likelihood function for the sensor be established. In theory, this requires specification of a probability distribution $P(\boldsymbol{z} \mid \boldsymbol{x}_{ij} = x_{ij})$ mapping each possible grid state x_{ij} to a distribution on

Avidbots Ex1016
Page 904 of 2154

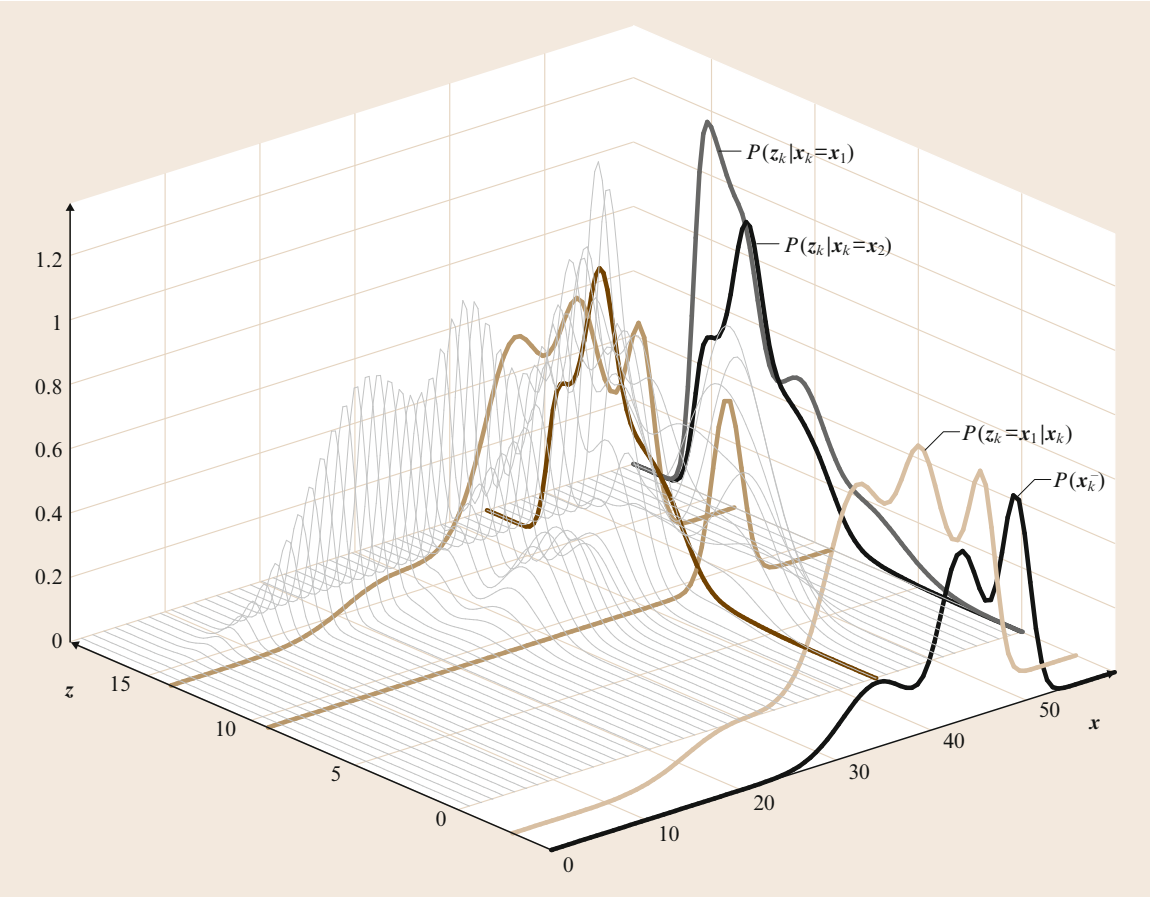


Fig. 35.2 Observation update for the full Bayes filter. Prior to observation, an observation model in the form of the conditional density $P(z_k \mid \boldsymbol{x}_k)$ is established. For a fixed value of $\boldsymbol{x}_k$, equal to $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$ for example, a density function $P(z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_1)$ or $P(z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_2)$ is defined describing the likelihood of making the observation z_k. Together the density $P(z_k \mid \boldsymbol{x}_k)$ is then a function of both z_k and $\boldsymbol{x}_k$. This conditional density then defines the observation model. Now, in operation, a specific observation $z_k = \boldsymbol{x}_1$ is made and the resulting distribution $P(z_k = \boldsymbol{x}_1 \mid \boldsymbol{x}_k)$ defines a density function (now termed the likelihood function) on $\boldsymbol{x}_k$. This density is then multiplied by the prior density $P(\boldsymbol{x}_k^-)$ and normalized to obtain the posterior distribution $P(\boldsymbol{x}_k \mid z_k)$ describing knowledge in the state after observation

observations. Practically, however, this is implemented simply as another *observation* grid so that for a specific observation $\boldsymbol{z} = z$ (taken from a specific location), a grid of likelihoods on the states $\boldsymbol{x}_{ij}$ is produced in the form $P(\boldsymbol{z} = z \mid \boldsymbol{x}_{ij}) = \Lambda(\boldsymbol{x}_{ij})$. It is then trivial to apply Bayes' rule to update the property value at each grid cell as

$$P^+(\boldsymbol{x}_{ij}) = C\Lambda(\boldsymbol{x}_{ij})P(\boldsymbol{x}_{ij}) \,, \quad \forall i, j \,, \tag{35.9}$$

where C us a normalizing constant obtained by summing posterior probabilities to one *at node ij only*. Computationally, this is a simple pointwise multiplication of two grids. Some care needs to be taken that the two grids appropriately overlap and align with each other at the right scale. In some instances it is also valuable to encode the fact that spatially adjacent cells will influence each other; that is, if we knew the value of the property (occupancy, temperature, for example) at ij we will have some belief also of the value of this property at adjacent nodes $i+1, j$, $i, j+1$, etc. Different sensors and the fusion of different sensor outputs is accommodated simply by building appropriate sensor models $\Lambda(\boldsymbol{x}_{ij})$.

Grids can also be used for tracking and self-tracking (localization). The state $\boldsymbol{x}_{ij}$ in this case is the location of the entity being tracked. This is a qualitatively different definition of state from that used in mapping. The probability $P(\boldsymbol{x}_{ij})$ must now be interpreted as the probability that the object being tracked occupies the grid cell ij. In the case of mapping, the sum of property probabilities at each grid cell is one, whereas in the case of tracking, the

Avidbots Ex1016
Page 905 of 2154

sum of location probabilities over *the whole grid* must sum to one. Otherwise, the procedure for updating is very similar. An observation grid is constructed which, when instantiated with an observation value, provides a location likelihood grid

$$P(\boldsymbol{z} = z \mid \boldsymbol{x}_{ij}) = \Lambda(\boldsymbol{x}_{ij}) \, .$$

Bayes' rule is then applied to update the location probability at each grid cell in the same form as (35.9) except that now the normalization constant C is obtained by summing posterior probabilities over *all* ij grid cells. This can become computationally expensive, especially if the grid has three or more dimensions. One major advantage of grid-based tracking is that it is easy to incorporate quite complex prior information. For example, if it is known that the object being tracked is on a road, then the probability location values for all off-road grid cells can simply be set to zero.

Grid-based fusion is appropriate to situations where the domain size and dimension are modest. In such cases, grid-based methods provide straightforward and effective fusion algorithms. Grid-based methods can be extended in a number of ways; to hierarchical (quad-tree) grids, or to irregular (triangular, pentagonal) grids. These can help reduce computation in larger spaces. Monte Carlo and particle filtering methods (Sect. 35.1.4) may be considered as grid-based methods, where the grid cells themselves are sample of the underlying probability density for the state.

35.1.3 The Kalman Filter

The Kalman filter is a recursive linear estimator that successively calculates an estimate for a continuous valued state, that evolves over time, on the basis of periodic observations of the state. The Kalman filter employs an explicit statistical model of how the parameter of interest $\boldsymbol{x}(t)$ evolves over time and an explicit statistical model of how the observations $\boldsymbol{z}(t)$ that are made are related to this parameter. The gains employed in a Kalman filter are chosen to ensure that, with certain assumptions about the observation and process models used, the resulting estimate $\hat{\boldsymbol{x}}(t)$ minimizes mean-squared error and is thus the conditional mean $\hat{\boldsymbol{x}}(t) = \mathrm{E}[\boldsymbol{x}(t) \mid \boldsymbol{Z}^t]$: an average, rather than a most likely value.

The Kalman filter has a number of features which make it ideally suited to dealing with complex multisensor estimation and data fusion problems. In particular, the explicit description of process and observations allows a wide variety of different sensor models to be incorporated within the basic algorithm. In addition, the consistent use of statistical measures of uncertainty makes it possible to quantitatively evaluate the role each sensor plays in overall system performance. Further, the linear recursive nature of the algorithm ensures that its application is simple and efficient. For these reasons, the Kalman filter has found widespread application in many different data fusion problems [35.6–9].

In robotics, the Kalman filter is most suited to problems in tracking, localization, and navigation, and less so to problems in mapping. This is because the algorithm works best with well-defined state descriptions (positions, velocities, for example), and for states where observation and time-propagation models are also well understood.

Observation and Transition Models

The Kalman filter may be considered a specific instance of the recursive Bayesian filter of (35.7) and (35.8) for the case where the probability densities on states are Gaussian. The starting point for the Kalman Filter algorithm is to define a model for the states to be estimated in the standard state-space form

$$\dot{\boldsymbol{x}}(t) = \mathbf{F}(t)\boldsymbol{x}(t) + \mathbf{B}(t)\boldsymbol{u}(t) + \mathbf{G}(t)\boldsymbol{v}(t) \, , \qquad (35.10)$$

where $\boldsymbol{x}(t)$ is the state vector of interest, $\boldsymbol{u}(t)$ is a known control input, $\boldsymbol{v}(t)$ is a random variable describing uncertainty in the evolution of the state, and $\mathbf{F}(t)$, $\mathbf{B}(t)$, and $\mathbf{G}(t)$ are matrices describing the contribution of states, controls, and noise to state transition [35.7]. An observation (output) model is also defined in standard state-space form

$$\boldsymbol{z}(t) = \mathbf{H}(t)\boldsymbol{x}(t) + \mathbf{D}(t)\boldsymbol{w}(t) \, , \qquad (35.11)$$

where $\boldsymbol{z}(t)$ is the observation vector, $\boldsymbol{w}(t)$ is a random variable describing uncertainty in the observation, and where $\mathbf{H}(t)$ and $\mathbf{D}(t)$ are matrices describing the contribution of state and noise to the observation.

These equations define the evolution of a continuous-time system with continuous observations being made of the state. However, the Kalman Filter is almost always implemented in discrete time $t_k = k$. It is straightforward [35.8] to obtain a discrete-time version of (35.10) and (35.11) in the form

$$\boldsymbol{x}(k) = \mathbf{F}(k)\boldsymbol{x}(k-1) + \mathbf{B}(k)\boldsymbol{u}(k) + \mathbf{G}(k)\boldsymbol{v}(k) \, , \qquad (35.12)$$

$$\boldsymbol{z}(k) = \mathbf{H}(k)\boldsymbol{x}(k) + \mathbf{D}(k)\boldsymbol{w}(k) \, . \qquad (35.13)$$

A basic assumption in the derivation of the Kalman filter is that the random sequences $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ describing process and observation noise are all Gaussian, temporally uncorrelated, and zero-mean

$$\mathrm{E}[\boldsymbol{v}(k)] = \mathrm{E}[\boldsymbol{w}(k)] = \mathbf{0} \, , \quad \forall k \, , \qquad (35.14)$$

Avidbots Ex1016
Page 906 of 2154

with known covariance

$$\mathrm{E}[\boldsymbol{v}(i)\boldsymbol{v}^{\mathrm{T}}(j)] = \delta_{ij}\mathbf{Q}(i)\,, \quad \mathrm{E}[\boldsymbol{w}(i)\boldsymbol{w}^{\mathrm{T}}(j)] = \delta_{ij}\mathbf{R}(i)\,. \tag{35.15}$$

It is also generally assumed that the process and observation noises are also uncorrelated

$$\mathrm{E}[\boldsymbol{v}(i)\boldsymbol{w}^{\mathrm{T}}(j)] = \mathbf{0}\,, \quad \forall i,j\,. \tag{35.16}$$

These are equivalent to a Markov property requiring observations and successive states to be conditionally independent. If the sequences $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ are temporally correlated, a shaping filter can be used to whiten the observations, again making the assumptions required for the Kalman filter valid [35.8]. If the process and observation noise sequences are correlated, then this correlation can also be accounted for in the Kalman filter algorithm [35.10]. If the sequence is not Gaussian, but is symmetric with finite moments, then the Kalman filter will still produce good estimates. If however, the sequence has a distribution which is skewed or otherwise pathological, results produced by the Kalman filter will be misleading and there will be a good case for using a more-sophisticated Bayesian filter [35.5].

Filtering Algorithm

The Kalman filter algorithm produces estimates that minimize mean-squared estimation error conditioned on a given observation sequence and so is the conditional mean

$$\hat{\boldsymbol{x}}(i\,|\,j) \triangleq \mathrm{E}[\boldsymbol{x}(i)\,|\,z(1),\cdots,z(j)] \triangleq \mathrm{E}[\boldsymbol{x}(i)\,|\,\boldsymbol{Z}^j]\,. \tag{35.17}$$

The estimate variance is defined as the mean-squared error in this estimate

$$\boldsymbol{P}(i\,|\,j) \triangleq \mathrm{E}\{[\boldsymbol{x}(i)-\hat{\boldsymbol{x}}(i\,|\,j)][\boldsymbol{x}(i)-\hat{\boldsymbol{x}}(i\,|\,j)]^T\,|\,\boldsymbol{Z}^j\}\,. \tag{35.18}$$

The estimate of the state at a time k given all information up to time k is written $\hat{\boldsymbol{x}}(k\,|\,k)$. The estimate of the state at a time k given only information up to time $k-1$ is called a one-step-ahead prediction (or just a prediction) and is written $\hat{\boldsymbol{x}}(k\,|\,k-1)$.

The Kalman filter algorithm is now stated without proof. Detailed derivations can be found in many books on the subject, for example [35.7, 8]. The state is assumed to evolve in time according to (35.12). Observations of this state are made at regular time intervals according to (35.13). The noise processes entering the system are assumed to obey (35.14–35.16). It is also assumed that an estimate $\hat{\boldsymbol{x}}(k-1\,|\,k-1)$ of the state $\boldsymbol{x}(k-1)$ at time $k-1$ based on all observations made up to and including time $k-1$ is available, and that this estimate is equal to the conditional mean of the true state $\boldsymbol{x}(k-1)$ conditioned on these observations. The conditional variance $\boldsymbol{P}(k-1\,|\,k-1)$ in this estimate is also assumed known. The Kalman filter then proceeds recursively in two stages (Fig. 35.3).

Prediction. A prediction $\hat{\boldsymbol{x}}(k\,|\,k-1)$ of the state at time k and its covariance $\boldsymbol{P}(k\,|\,k-1)$ is computed according to

$$\hat{\boldsymbol{x}}(k\,|\,k-1) = \mathbf{F}(k)\hat{\boldsymbol{x}}(k-1\,|\,k-1) + \mathbf{B}(k)\boldsymbol{u}(k)\,, \tag{35.19}$$

$$\boldsymbol{P}(k\,|\,k-1) = \mathbf{F}(k)\boldsymbol{P}(k-1\,|\,k-1)\mathbf{F}^{\mathrm{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{G}^{\mathrm{T}}(k)\,. \tag{35.20}$$

Update. At time k an observation $\boldsymbol{z}(k)$ is made and the updated estimate $\hat{\boldsymbol{x}}(k\,|\,k)$ of the state $\boldsymbol{x}(k)$, together with the updated estimate covariance $\boldsymbol{P}(k\,|\,k)$, is computed from the state prediction and observation according to

$$\hat{\boldsymbol{x}}(k\,|\,k) = \hat{\boldsymbol{x}}(k\,|\,k-1) + \mathbf{W}(k)[\boldsymbol{z}(k) - \mathbf{H}(k)\hat{\boldsymbol{x}}(k\,|\,k-1)]\,, \tag{35.21}$$

$$\boldsymbol{P}(k\,|\,k) = \boldsymbol{P}(k\,|\,k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^{\mathrm{T}}(k)\,, \tag{35.22}$$

where the gain matrix $\mathbf{W}(k)$ is given by

$$\mathbf{W}(k) = \boldsymbol{P}(k\,|\,k-1)\mathbf{H}(k)\mathbf{S}^{-1}(k)\,, \tag{35.23}$$

where

$$\mathbf{S}(k) = \mathbf{R}(k) + \mathbf{H}(k)\boldsymbol{P}(k\,|\,k-1)\mathbf{H}(k) \tag{35.24}$$

is the innovation covariance. The difference between the observation $\boldsymbol{z}(k)$ and the predicted observation $\mathbf{H}(k)\hat{\boldsymbol{x}}(k\,|\,k-1)$ is termed the *innovation* or residual $\boldsymbol{v}(k)$

$$\boldsymbol{v}(k) = \boldsymbol{z}(k) - \mathbf{H}(k)\hat{\boldsymbol{x}}(k\,|\,k-1)\,. \tag{35.25}$$

The innovation is an important measure of the deviation between the filter estimates and the observation sequence. Indeed, because the true states are not usually available for comparison with the estimated states, the innovation is often the only measure of how well the estimator is performing. The innovation is particularly important in data association.

The Extended Kalman Filter

The extended Kalman filter (EKF) is a form of the Kalman filter that can be employed when the state model and/or the observation model are nonlinear. The EKF is briefly described in this section.

Avidbots Ex1016
Page 907 of 2154

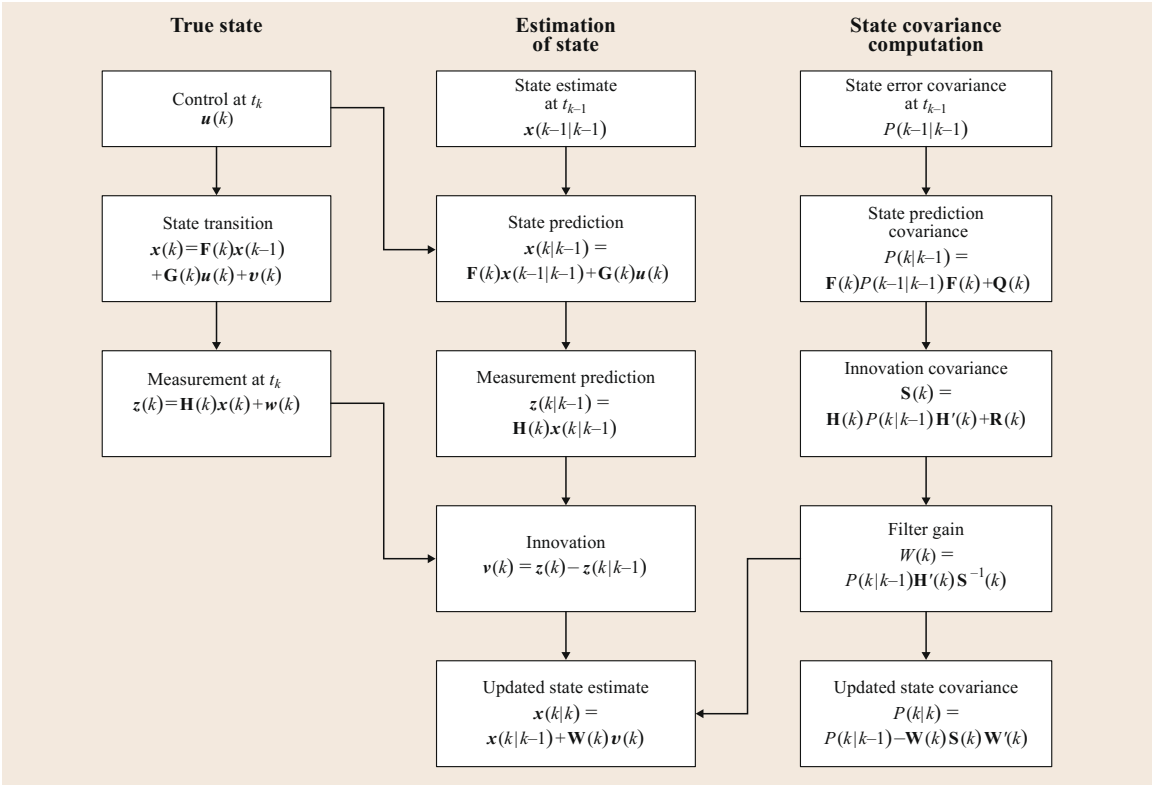


Fig. 35.3 Block diagram of the Kalman filter cycle (after *Bar-Shalom* and *Fortmann* [35.7])

The state models considered by the EKF are described in state-space notation by a first-order nonlinear vector differential equation or state model of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}[\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), t] \,, \tag{35.26}$$

where $\boldsymbol{f}[\cdot,\cdot,\cdot,\cdot]$ is now a general nonlinear mapping of state and control input to state transition. The observation models considered by the EKF are described in state-space notation by a nonlinear vector function in the form

$$\boldsymbol{z}(t) = \boldsymbol{h}[\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{w}(t), t] \,, \tag{35.27}$$

where $\boldsymbol{h}[\cdot,\cdot,\cdot,\cdot]$ is now a general nonlinear mapping of state and control input to observations.

The EKF, like the Kalman filter, is almost always implemented in discrete time. By integration and with appropriate identification of discrete time states and observations, the state model is written

$$\boldsymbol{x}(k) = \boldsymbol{f}[\boldsymbol{x}(k-1), \boldsymbol{u}(k), \boldsymbol{v}(k), k] \,, \tag{35.28}$$

and the observation model as

$$\boldsymbol{z}(k) = \boldsymbol{h}[\boldsymbol{x}(k), \boldsymbol{w}(k)] \,. \tag{35.29}$$

Like the Kalman filter, it is assumed that the noises $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ are all Gaussian, temporally uncorrelated, and zero-mean with known variance as defined in (35.14–35.16). The EKF aims to minimize the mean-squared error and thereby compute an approximation to the conditional mean. It is assumed therefore that an estimate of the state at time $k-1$ is available which is approximately equal to the conditional mean, $\hat{\boldsymbol{x}}(k-1 \mid k-1) \approx \mathrm{E}[\boldsymbol{x}(k-1) \mid Z^{k-1}]$. The EKF algorithm will now be stated without proof. Detailed derivations may be found in any number of books on the subject. The principle stages in the derivation of the EKF follow directly from those of the linear Kalman filter with the additional step that the process and observation models are linearized as a Taylor series about the estimate and prediction, respectively. The algorithm has two stages: prediction and update.

Prediction. A prediction $\hat{\boldsymbol{x}}(k \mid k-1)$ of the state at time k and its covariance $\boldsymbol{P}(k \mid k-1)$ is computed according to

$$\hat{\boldsymbol{x}}(k \mid k-1) = \boldsymbol{f}[\hat{\boldsymbol{x}}(k-1 \mid k-1), \boldsymbol{u}(k)] \,, \tag{35.30}$$

$$\boldsymbol{P}(k \mid k-1) = \nabla \boldsymbol{f}_{\boldsymbol{x}}(k) \boldsymbol{P}(k-1 \mid k-1) \nabla^{\mathrm{T}} \boldsymbol{f}_{\boldsymbol{x}}(k) + \nabla \boldsymbol{f}_{\boldsymbol{v}}(k) \mathbf{Q}(k) \nabla^{\mathrm{T}} \boldsymbol{f}_{\boldsymbol{v}}(k) \,. \tag{35.31}$$

Avidbots Ex1016
Page 908 of 2154

Update. At time k an observation $\boldsymbol{z}(k)$ is made and the updated estimate $\hat{\boldsymbol{x}}(k \mid k)$ of the state $\boldsymbol{x}(k)$, together with the updated estimate covariance $\boldsymbol{P}(k \mid k)$ is computed from the state prediction and observation according to

$$\hat{\boldsymbol{x}}(k \mid k) = \hat{\boldsymbol{x}}(k \mid k-1) + \mathbf{W}(k)\{\boldsymbol{z}(k) - \boldsymbol{h}[\hat{\boldsymbol{x}}(k \mid k-1)]\} , \quad (35.32)$$

$$\boldsymbol{P}(k \mid k) = \boldsymbol{P}(k \mid k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^{\mathrm{T}}(k) , \quad (35.33)$$

where

$$\mathbf{W}(k) = \boldsymbol{P}(k \mid k-1)\nabla^{\mathrm{T}}\boldsymbol{h}_{\boldsymbol{x}}(k)\mathbf{S}^{-1}(k) , \quad (35.34)$$

and

$$\mathbf{S}(k) = \nabla\boldsymbol{h}_{w}(k)\mathbf{R}(k)\nabla^{\mathrm{T}}\boldsymbol{h}_{w}(k) + \nabla\boldsymbol{h}_{\boldsymbol{x}}(k)\boldsymbol{P}(k \mid k-1)\nabla^{\mathrm{T}}\boldsymbol{h}_{\boldsymbol{x}}(k) , \quad (35.35)$$

and where the Jacobian $\nabla\boldsymbol{f}_{\cdot}(k)$ is evaluated at $\boldsymbol{x}(k-1) = \hat{\boldsymbol{x}}(k-1 \mid k-1)$ and $\nabla\boldsymbol{h}_{\cdot}(k)$ is evaluated at $\boldsymbol{x}(k) = \hat{\boldsymbol{x}}(k \mid k-1)$.

A comparison of (35.19–35.24) with (35.30–35.35) makes it clear that the EKF algorithm is very similar to the linear Kalman filter algorithm, with the substitutions $\mathbf{F}(k) \to \nabla\boldsymbol{f}_{\boldsymbol{x}}(k)$ and $\mathbf{H}(k) \to \nabla\boldsymbol{h}_{\boldsymbol{x}}(k)$ being made in the equations for the variance and gain propagation. Thus, the EKF is, in effect, a linear estimator for a state error which is described by a *linear* equation and which is being observed according to a *linear* equation of the form of (35.13).

The EKF works in much the same way as the linear Kalman filter with some notable caveats:

- The Jacobians $\nabla\boldsymbol{f}_{\boldsymbol{x}}(k)$ and $\nabla\boldsymbol{h}_{\boldsymbol{x}}(k)$ are typically not constant, being functions of both state and time step. This means that unlike the linear filter, the covariances and gain matrix must be computed online as estimates and predictions are made available, and will not in general tend to constant values. This significantly increases the amount of computation which must be performed online by the algorithm.
- As the linearized model is derived by perturbing the true state and observation models around a predicted or nominal trajectory, great care must be taken to ensure that these predictions are always close enough to the true state that second order terms in the linearization are indeed insignificant. If the nominal trajectory is too far away from the true trajectory then the true covariance will be much larger than the estimated covariance and the filter will become poorly matched. In extreme cases the filter may also become unstable.
- The EKF employs a linearized model that must be computed from an approximate knowledge of the state. Unlike the linear algorithm this means that the filter must be accurately initialized at the start of operation to ensure that the linearized models obtained are valid. If this is not done, the estimates computed by the filter will simply be meaningless.

The Information Filter

The information filter is mathematically equivalent to a Kalman filter. However, rather than generating state estimates $\hat{\boldsymbol{x}}(i \mid j)$ and covariances $\boldsymbol{P}(i \mid j)$ it uses information state variables $\hat{\boldsymbol{y}}(i \mid j)$ and information matrices $\mathbf{Y}(i \mid j)$, which are related to each other through the relationships

$$\hat{\boldsymbol{y}}(i \mid j) = \boldsymbol{P}^{-1}(i \mid j)\hat{\boldsymbol{x}}(i \mid j) ,$$
$$\mathbf{Y}(i \mid j) = \boldsymbol{P}^{-1}(i \mid j) . \quad (35.36)$$

The information filter has the same prediction–update structure as the Kalman filter.

Prediction. A prediction $\hat{\boldsymbol{y}}(k \mid k-1)$ of the information state at time k and its information matrix $\mathbf{Y}(k \mid k-1)$ is computed according to (Joseph form [35.8])

$$\hat{\boldsymbol{y}}(k \mid k-1) = (\mathbf{1} - \boldsymbol{\Omega}\mathbf{G}^{\mathrm{T}})\mathbf{F}^{-\mathrm{T}}\hat{\boldsymbol{y}}(k-1 \mid k-1) + \mathbf{Y}(k \mid k-1)\mathbf{B}\boldsymbol{u}(k) , \quad (35.37)$$

$$\mathbf{Y}(k \mid k-1) = \mathbf{M}(k) - \boldsymbol{\Omega}\boldsymbol{\Sigma}\boldsymbol{\Omega}^{\mathrm{T}} , \quad (35.38)$$

respectively, where

$$\mathbf{M}(k) = \mathbf{F}^{-\mathrm{T}}\mathbf{Y}(k-1 \mid k-1)\mathbf{F}^{-1} ,$$
$$\boldsymbol{\Sigma} = \mathbf{G}^{\mathrm{T}}\mathbf{M}(k)\mathbf{G} + \mathbf{Q}^{-1} ,$$

and

$$\boldsymbol{\Omega} = \mathbf{M}(t_k)\mathbf{G}\boldsymbol{\Sigma}^{-1} .$$

It should be noted that $\boldsymbol{\Sigma}$, whose inverse is required to compute $\boldsymbol{\Omega}$, has the same dimensionality as the process driving noise, which is normally considerably smaller than the state dimension. Further, the matrix $\mathbf{F}^{-1}$ is the state-transition matrix evaluated backwards in time and so must always exist.

Update. At time k an observation $\boldsymbol{z}(k)$ is made and the updated information state estimate $\hat{\boldsymbol{y}}(k \mid k)$ together with the updated information matrix $\mathbf{Y}(k \mid k)$ is computed from

$$\hat{\boldsymbol{y}}(k \mid k) = \hat{\boldsymbol{y}}(k \mid k-1) + \mathbf{H}(k)\mathbf{R}^{-1}(k)\boldsymbol{z}(k) , \quad (35.39)$$

$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \mathbf{H}(k)\mathbf{R}^{-1}(k)\mathbf{H}^{\mathrm{T}}(k) . \quad (35.40)$$

Avidbots Ex1016
Page 909 of 2154

We emphasize that (35.38) and (35.37) are mathematically identical to (35.19) and (35.20), and that (35.39) and (35.40) are mathematically identical to (35.21) and (35.22). It will be noted that there is a duality between information and state space forms [35.10]. This duality is evident from the fact that $\mathbf{\Omega}$ and $\mathbf{\Sigma}$ in the prediction stage of the information filter play an equivalent role to the gain matrix $\mathbf{W}$ and innovation covariance $\mathbf{S}$ in the update stage of the Kalman filter. Furthermore, the simple linear update step for the information filter is mirrored in the simple linear prediction step for the Kalman filter.

The main advantage of the information filter over the Kalman filter in data fusion problems is the relative simplicity of the update stage. For a system with n sensors, the fused information state update is *exactly* the linear sum of information contributions from all sensors as

$$\hat{\boldsymbol{y}}(k \mid k) = \hat{\boldsymbol{y}}(k \mid k-1) + \sum_{i=1}^{n} \mathbf{H}_i(k)\mathbf{R}_i^{-1}(k)\boldsymbol{z}_i(k) \,,$$
$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \sum_{i=1}^{n} \mathbf{H}_i(k)\mathbf{R}_i^{-1}(k)\mathbf{H}_i^{\mathrm{T}}(k) \,. \tag{35.41}$$

The reason such an expression exists in this form is that the information filter is essentially a log-likelihood expression of Bayes' rule, where products of likelihoods (35.4) are turned into sums. No such simple expression for multisensor updates exists for the Kalman filter. This property of the information filter has been exploited for data fusion in robotic networks [35.11, 12] and more recently in robot navigation and localization problems [35.1]. One substantial disadvantage of the information filter is the coding of nonlinear models, especially for the prediction step.

When to Use a Kalman or Information Filter

Kalman or information filters are appropriate to data fusion problems where the entity of interest is well defined by a continuous parametric state. This would include estimation of the position, attitude, and velocity of a robot or other object, or the tracking of a simple geometric feature such as a point, line or curve. Kalman and information filters are inappropriate for estimating properties such as spatial occupancy, discrete labels, or processes whose error characteristics are not easily parameterized.

35.1.4 Sequential Monte Carlo Methods

Monte Carlo (MC) filter methods describe probability distributions as a set of weighted samples of an underlying state space. MC filtering then uses these samples to simulate probabilistic inference usually through Bayes' rule. Many samples or simulations are performed. By studying the statistics of these samples as they progress through the inference process, a probabilistic picture of the process being simulated can be built up.

Representing Probability Distributions

In sequential Monte Carlo methods, probability distributions are described in terms of a set of support points (state-space values) $\boldsymbol{x}^i$, $i = 1, \cdots, N$, together with a corresponding set of normalized weights w^i, $i = 1, \cdots, N$, where $\sum_i w^i = 1$. The support points and weights can be used to define a probability density function in the form

$$P(\boldsymbol{x}) \approx \sum_{i=1}^{N} w^i \delta(\boldsymbol{x} - \boldsymbol{x}^i) \,. \tag{35.42}$$

A key question is how these support points and weights are selected to obtain a faithful representation of the probability density $P(\boldsymbol{x})$. The most general way of selecting support values is to use an *importance density* $q(\boldsymbol{x})$. The support values $\boldsymbol{x}^i$ are drawn as samples from this density; where the density has high probability, more support values are chosen, and where the density has low probability, few support support vectors are selected. The weights in (35.42) are then computed from

$$w^i \propto \frac{P(\boldsymbol{x}^i)}{q(\boldsymbol{x}^i)} \,. \tag{35.43}$$

Practically, a sample $\boldsymbol{x}^i$ is drawn from the importance distribution. The sample is then instantiated in the underlying probability distribution to yield the value $P(\boldsymbol{x} = \boldsymbol{x}^i)$. The ratio of the two probability values, appropriately normalized, then becomes the weight.

There are two instructive extremes of the importance sampling method:

1. At one extreme, the importance density could be taken to be a uniform distribution, so the support values $\boldsymbol{x}^i$ are uniformly distributed on the state space in a close approximation to a grid. The probabilities $q(\boldsymbol{x}^i)$ are also therefore equal. The weights computed from (35.43) are then simply proportional to the probabilities $w^i \propto P(\boldsymbol{x} = \boldsymbol{x}^i)$. The result is a model for the distribution which looks very like the regular grid model.
2. At the other extreme, we could choose an importance density equal to the probability model $q(\boldsymbol{x}) = P(\boldsymbol{x})$. Samples of the support values $\boldsymbol{x}^i$ are now drawn from this density. Where the density is high

Avidbots Ex1016
Page 910 of 2154

there will be many samples, where the density is low there will be few samples. However, if we substitute $q(\boldsymbol{x}^i) = P(\boldsymbol{x}^i)$ into (35.43), it is clear that the weights all become equal $w^i = \frac{1}{N}$. A set of samples with equal weights is known as a *particle* distribution.

It is, of course, possible to mix these two representations to describe a probability distribution both in terms of a set of weights and in terms of a set of support values. The complete set of samples and weights describing a probability distribution $\{\boldsymbol{x}^i, w^i\}_{i=1}^N$ is termed a *random measure*.

The Sequential Monte Carlo Method

Sequential Monte Carlo (SMC) filtering is a simulation of the recursive Bayes update equations using sample support values and weights to describe the underlying probability distributions.

The starting point is the recursive or sequential Bayes observation update given in (35.7) and (35.8). The SMC recursion begins with an a posterior probability density represented by a set of support values and weights $\{\boldsymbol{x}_{k-1}^i, w_{k-1|k-1}^i\}_{i=1}^{N_{k-1}}$ in the form

$$P\left(\boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}\right) = \sum_{i=1}^{N_{k-1}} w_{k-1}^i \delta\left(\boldsymbol{x}_{k-1} - \boldsymbol{x}_{k-1}^i\right) . \tag{35.44}$$

The prediction step requires that (35.44) is substituted into (35.8) where the joint density is marginalized. Practically however, this complex step is avoided by implicitly assuming that the importance density is exactly the transition model as

$$q_k\left(\boldsymbol{x}_k^i\right) = P\left(\boldsymbol{x}_k^i \mid \boldsymbol{x}_{k-1}^i\right) . \tag{35.45}$$

This allows new support values $\boldsymbol{x}_k^i$ to be drawn on the basis of old support values $\boldsymbol{x}_{k-1}^i$ while leaving the weights unchanged $w_k^i = w_{k-1}^i$. With this, the prediction becomes

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}\right) = \sum_{i=1}^{N_k} w_{k-1}^i \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^i\right) . \tag{35.46}$$

The SMC observation update step is relatively straightforward. An observation model $P(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$ is defined. This is a function on both variables, $\boldsymbol{z}_k$ and $\boldsymbol{x}_k$, and is a probability distribution on $\boldsymbol{z}_k$ (integrates to unity). When an observation or measurement is made, $\boldsymbol{z}_k = z_k$, the observation model becomes a function of state $\boldsymbol{x}_k$ only. If samples of the state are taken $\boldsymbol{x}_k = \boldsymbol{x}_k^i$, $i = 1 \cdots, N_k$, the observation model $P(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i)$ becomes a set of scalars describing the likelihood that the sample $\boldsymbol{x}_k^i$ could have given rise to the observation z_k. Substituting these likelihoods and (35.46) into (35.7) gives

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k\right) = C \sum_{i=1}^{N_k} w_{k-1}^i P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i\right) \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^i\right) . \tag{35.47}$$

This is normally implemented in the form of an updated set of normalized weights

$$w_k^i = \frac{w_{k-1}^i P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i\right)}{\sum_{j=1}^{N_k} w_{k-1}^j P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^j\right)} , \tag{35.48}$$

and so

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k\right) = \sum_{i=1}^{N_k} w_k^i \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^i\right) . \tag{35.49}$$

Note that the support values in (35.49) are the same as those in (35.46), only the weights have been changed by the observation update.

The implementation of the SMC method requires the enumeration of models for both the state transition $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ and the observation $P(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$. These need to be presented in a form that allows instantiation of values for $\boldsymbol{z}_k$, $\boldsymbol{x}_k$, and $\boldsymbol{x}_{k-1}$. For low-dimensional state spaces, interpolation in a lookup table is a viable representation. For high-dimensional state spaces, the preferred method is to provide a representation in terms of a function.

Practically, (35.46) and (35.49) are implemented as follows.

Time Update. A process model is defined in the usual state-space form as $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1}, k)$, where $\boldsymbol{w}_k$ is an independent noise sequence with known probability density $P(\boldsymbol{w}_k)$. The prediction step is now implemented as follows: N_k samples $\boldsymbol{w}_k^i$, $i = 1, \cdots, N_k$ are drawn from the distribution $P(\boldsymbol{w}_k)$. The N_k support values $\boldsymbol{x}_{k-1}^i$ together with the samples $\boldsymbol{w}_k^i$ are passed through the process model as

$$\boldsymbol{x}_k^i = \boldsymbol{f}\left(\boldsymbol{x}_{k-1}^i, \boldsymbol{w}_{k-1}^i, k\right) , \tag{35.50}$$

yielding a new set of support vectors $\boldsymbol{x}_k^i$. The weights for these support vectors w_{k-1}^i are not changed. In effect, the process model is simply used to do N_k simulations of state propagation.

Avidbots Ex1016
Page 911 of 2154

Observation Update. The observation model is also defined in the usual state-space form as $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{v}_k, k)$, where $\boldsymbol{v}_k$ is an independent noise sequence with known probability density $P(\boldsymbol{v}_k)$. The observation step is now implemented as follows. A measurement $\boldsymbol{z}_k = z_k$ is made. For each support value $\boldsymbol{x}_k^i$, a likelihood is computed as

$$\Lambda\left(\boldsymbol{x}_k^i\right) = P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i\right) . \tag{35.51}$$

Practically, this requires that the observation model be in an equational form (such as a Gaussian) which allows computation of the likelihood in the error between the measured value z_k and the observations predicted by each particle $\boldsymbol{h}(\boldsymbol{x}_k^i, k)$. The updated weights after observation are just

$$w_k^i \propto w_{k-1}^i P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k^i\right) . \tag{35.52}$$

Resampling

After the weights are updated it is usual to resample the measure $\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N$. This focuses the samples in on those areas that have most probability density. The decision to resample is made on the basis of the *effective number*, N_{eff} of particles in the sample, approximately estimated from

$$N_{\text{eff}} = \frac{1}{\sum_i (w_k^i)^2} .$$

The sampling importance resampling (SIR) algorithm resamples at every cycle so that the weights are always equal. One of the key problems with resampling is that the sample set fixates on a few highly likely samples. This problem of fixating on a few highly likely particles during resampling is known as *sample impoverishment*. Generally, it is good to resample when N_{eff} falls to some fraction of the actual samples (say $1/2$).

When to Use Monte Carlo Methods

Monte Carlo (MC) methods are well suited to problems where state-transition models and observation models are highly nonlinear. This is because sample-based methods can represent very general probability densities. In particular multimodal or multiple hypothesis density functions are well handled by Monte Carlo techniques. One caveat to note however is that the models $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ and $P(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$ must be enumerable in all cases and must typically be in a simple parametric form. MC methods also span the gap between parametric and grid-based data fusion methods.

Monte Carlo methods are inappropriate for problems where the state space is of high dimension. In general the number of samples required to faithfully model a given density increases exponentially with state-space dimension. The effects of dimensionality can be limited by marginalizing out states that can be modeled without sampling, a procedure known as Rao–Blackwellization.

35.1.5 Alternatives to Probability

The representation of uncertainty is so important to the problem of information fusion that a number of alternative modeling techniques have been proposed to deal with perceived limitations in probabilistic methods.

There are three main perceived limitations of probabilistic modeling techniques:

1. Complexity: the need to specify a large number of probabilities to be able to apply probabilistic reasoning methods correctly.
2. Inconsistency: the difficulties involved in specifying a consistent set of beliefs in terms of probability and using these to obtain consistent deductions about states of interest.
3. Precision of models: the need to be precise in the specification of probabilities for quantities about which little is known.
4. Uncertainty about uncertainty: the difficulty in assigning probability in the face of uncertainty, or ignorance about the source of information.

There are three main techniques put forward to address these issues: interval calculus, fuzzy logic, and the theory of evidence (Dempster–Shafer methods). We briefly discuss each of these in turn.

Interval Calculus

The representation of uncertainty using an interval to bound true parameter values has a number of potential advantages over probabilistic techniques. In particular, intervals provide a good measure of uncertainty in situations where there is a lack of probabilistic information, but in which sensor and parameter error is known to be bounded. In interval techniques, the uncertainty in a parameter x is simply described by a statement that the true value of the state x is known to be bounded from below by a, and from above by b; $x \in [a, b]$. It is important that no other additional probabilistic structure is implied; in particular the statement $x \in [a, b]$ does not necessarily imply that x is equally probable (uniformly distributed) over the interval $[a, b]$.

There are a number of simple and basic rules for the manipulation of interval errors. These are described in detail in the book by *Moore* [35.13] (whose analysis was originally aimed at understanding limited-precision computer arithmetic). Briefly, with $a, b, c, d \in \mathbb{R}$, addi-

Avidbots Ex1016
Page 912 of 2154

tion, subtraction, multiplication, and division are defined by the following algebraic relations

$$[a,b]+[c,d]=[a+c,b+d]\,,$$
$$[a,b]-[c,d]=[a-d,b-c]\,, \tag{35.53}$$
$$[a,b]\times[c,d]=[\min(ac,ad,bc,bd),\ \max(ac,ad,bc,bd)]\,, \tag{35.54}$$
$$\frac{[a,b]}{[c,d]}=[a,b]\times\left[\frac{1}{d},\frac{1}{c}\right]\,,\quad 0\notin[c,d]\,. \tag{35.55}$$

It can be shown that interval addition and multiplication are both associative and commutative. Interval arithmetic admits an obvious metric distance measure

$$d([a,b],[c,d])=\max(|a-c|,|b-d|)\,. \tag{35.56}$$

Matrix arithmetic using intervals is also possible, but substantially more complex, particularly when matrix inversion is required.

Interval calculus methods are sometimes used for detection. However, they are not generally used in data fusion problems since:

1. It is difficult to get results that converge to anything of value (it is too pessimistic), and
2. It is hard to encode dependencies between variables which are at the core of many data fusion problems.

Fuzzy Logic

Fuzzy logic has found widespread popularity as a method for representing uncertainty particularly in applications such as supervisory control and high-level data fusion tasks. It is often claimed that fuzzy logic provides an ideal tool for inexact reasoning, particularly in rule-based systems. Certainly, fuzzy logic has had some notable success in practical application.

A great deal has been written about fuzzy sets and fuzzy logic (e.g. [35.14] and the discussion in [35.15, Chap. 11]). Here we briefly describe the main definitions and operations without any attempt to consider the more advanced features of fuzzy-logic methods.

Consider a universal set consisting of the elements x; $\mathcal{X}=\{x\}$. Consider a proper subset $\mathcal{A}\subseteq\mathcal{X}$ such that

$$\mathcal{A}=\{x\mid x \text{ has some specific property}\}\,.$$

In conventional logic systems, we can define a membership function $\mu_A(x)$ (also called the *characteristic function* which reports if a specific element $x\in\mathcal{X}$ is a member of this set

$$\mathcal{A}\rightleftharpoons\mu_A(x)=\begin{cases}1 & \text{if } x\in A\\ 0 & \text{if } x\notin A\end{cases}\,.$$

For example $\mathcal{X}$ may be the set of all aircraft. The set $\mathcal{A}$ may be the set of all supersonic aircraft. In the fuzzy logic literature, this is known as a *crisp* set. In contrast, a fuzzy set is one in which there is a *degree of membership*, ranging between 0 and 1. A fuzzy membership function $\mu_A(x)$ then defines the degree of membership of an element $x\in\mathcal{X}$ to the set $\mathcal{A}$. For example, if $\mathcal{X}$ is again the set of all aircraft, $\mathcal{A}$ may be the set of all *fast* aircraft. Then the fuzzy membership function $\mu_A(x)$ assigns a value between 0 and 1 indicating the degree of membership of every aircraft x to this set. Formally

$$\mathcal{A}\rightleftharpoons\mu_A\mapsto[0,1]\,.$$

Composition rules for fuzzy sets follow the composition processes for normal crisp sets, for example,

$$\mathcal{A}\cap\mathcal{B}\rightleftharpoons\mu_{A\cap B}(x)=\min[\mu_A(x),\mu_B(x)]\,,$$
$$\mathcal{A}\cup\mathcal{B}\rightleftharpoons\mu_{A\cup B}(x)=\max[\mu_A(x),\mu_B(x)]\,.$$

The normal properties associated with binary logic now hold: commutativity, associativity, idempotence, distributivity, De Morgan's law, and absorption. The only exception is that the law of the excluded middle is no longer true

$$\mathcal{A}\cup\overline{\mathcal{A}}\neq\mathcal{X}\,,\quad \mathcal{A}\cap\overline{\mathcal{A}}\neq\emptyset\,.$$

Together these definitions and laws provide a systematic means of reasoning about inexact values.

The relationship between fuzzy set theory and probability is still hotly debated.

Evidential Reasoning

Evidential reasoning (often called the Dempster–Shafer theory of evidence after the originators of these ideas) has seen intermittent success particularly in automated reasoning applications. Evidential reasoning is qualitatively different from either probabilistic methods or fuzzy set theory in the following sense. Consider a universal set $\mathcal{X}$. In probability theory or fuzzy set theory, a belief mass may be placed on any element $x_i\in\mathcal{X}$ and indeed on any subset $\mathcal{A}\subseteq\mathcal{X}$. In evidential reasoning, belief mass can not only be placed on elements and sets, but also sets of sets. Specifically, while the domain of probabilistic methods is all possible subsets $\mathcal{X}$, the domain of evidential reasoning is the power set $2^{\mathcal{X}}$.

As an example, consider the mutually exclusive set $\mathcal{X}=\{\text{occupied},\text{empty}\}$. In probability theory we

Avidbots Ex1016
Page 913 of 2154

might assign a probability to each possible event, for example, $P(\text{occupied}) = 0.3$, and thus $P(\text{empty}) = 0.7$. In evidential reasoning, we construct the set of all subsets

$$2^{\mathcal{X}} = \{\{\text{occupied}, \text{empty}\}, \{\text{occupied}\}, \{\text{empty}\}, \emptyset\} ,$$

and belief mass is assigned to all elements of this set as

$$\begin{aligned} m(\{\text{occupied}, \text{empty}\}) &= 0.5 , \\ m(\{\text{occupied}\}) &= 0.3 , \\ m(\{\text{empty}\}) &= 0.2 , \\ m(\emptyset) &= 0.0 , \end{aligned}$$

(the null set $\emptyset$ is assigned a belief mass of zero for normalization purposes). The interpretation of this is that there is a 30% chance of occupied, a 20% chance of empty, and a 50% chance of either occupied or empty. In effect, the measure placed on the set containing both occupied and empty is a measure of ignorance or inability to distinguish between the two alternatives. See [35.16] for a more-detailed example of applying the evidential method to certainty-grid navigation.

Evidential reasoning thus provides a method of capturing ignorance or an inability to distinguish between alternatives. In probability theory, this would be dealt with in a very different manner by assigning an equal or uniform probability to each alternative. Yet, stating that there is a 50% chance of occupancy is clearly *not* the same as saying that it is unknown whether it will occupied or not. The use of the power set as the *frame of discernment* allows a far richer representation of beliefs. However, this comes at the cost of a substantial increase in complexity. If there are n elements in the original set $\mathcal{X}$, then there will be 2^n possible subsets on which a belief mass will be assigned. For large n, this is clearly intractable. Further, when the set is continuous, the set of all subsets is not even measurable.

Evidential reasoning methods provide a means of assigning, and combining belief masses on sets. Methods also exist for obtaining related measures called *support* and *plausibility* which, in effect, provide upper and lower probability bounds in agreement with Dempster's original formulation of this method.

Evidential reasoning can play an important role in discrete data fusion systems, particularly in areas such as attribute fusion and situation assessment, where information may be unknown or ambiguous. Its use in lower-level data fusion problems is challenging as the assignment of belief mass to the power set scales exponentially with state cardinality.

35.2 Multisensor Fusion Architectures

The multisensor fusion methods described in the previous section provide the algorithmic means by which sensor data and their associated uncertainty models can be used to construct either implicit or explicit models of the environment. However, a multisensor fusion system must include many other functional components to manage and control the fusion process. The organization of these is termed a *multisensor fusion architecture*.

35.2.1 Architectural Taxonomy

Multisensor systems architectures can be organized in various ways. The military community has developed a layout of functional architectures based on the joint directors of the laboratories (JDL) model for multisensor systems. This approach views multisensor fusion in terms of signal, feature, threat and situation analysis levels (so-called JDL levels). The assessment of such systems is specified in terms of tracking performance, survivability, efficiency and bandwidth. Such measures are not generally appropriate in robotics applications and so the JDL model is not discuss this further here (see [35.17, 18] for details). Other classification schemes distinguish between low- and high-level fusion [35.19], or centralized versus decentralized processing or data versus variable [35.20].

A general architectural framework for multisensor robotic systems has been developed and described in detail by *Makarenko* [35.21], and we will base our discussion on this approach. A system architecture is defined as follows.

Meta-Architecture

A set of high-level considerations that strongly characterize the system structure. The selection and organization of the system elements may be guided by aesthetics, efficiency, or other design criteria and goals (for example, system and component comprehensibility, modularity, scalability, portability, interoperability, (de)centralization, robustness, fault tolerance).

Algorithmic Architecture

A specific set of information fusion and decision-making methods. These methods address data heterogeneity, registration, calibration, consistency, informa-

tion content, independence, time interval and scale, and relationships between models and uncertainty.

Conceptual Architecture

The granularity and functional roles of components (specifically, mappings from algorithmic elements to functional structures).

Logical Architecture

Detailed canonical component types (i. e., object-oriented specifications) and interfaces to formalize intercomponent services. Components may be ad hoc or regimented, and other concerns include granularity, modularity, reuse, verification, data structures, semantics, etc. Communication issues include hierarchical versus heterarchical organization, shared memory versus message passing, information-based characterizations of subcomponent interactions, pull/push mechanisms, subscribe–publish mechanisms, etc. Control involves both the control of actuation systems within the multisensor fusion system, as well as control of information requests and dissemination within the system, and any external control decisions and commands.

Execution Architecture

Defines mapping of components to execution elements. This includes internal or external methods of ensuring correctness of the code (i. e., that the environment and sensor models have been correctly transformed from mathematical or other formal descriptions into computer implementations), and also validation of the models (i. e., ensure that the formal descriptions match physical reality to the required extent).

In any closed-loop control system, sensors are used to provide the feedback information describing the current status of the system and its uncertainties. Building a sensor system for a given application is a system engineering process that includes the analysis of system requirements, a model of the environment, the determination of system behavior under different conditions, and the selection of suitable sensors [35.22]. The next step in building the sensor system is to assemble the hardware components and develop the necessary software modules for data fusion and interpretation. Finally, the system is tested, and the performance is analyzed. Once the system is built, it is necessary to monitor the different components of the system for the purpose of testing, debugging, and analysis. The system also requires quantitative measures in terms of time complexity, space complexity, robustness, and efficiency.

In addition, designing and implementing real-time systems are becoming increasingly complex owing to many added features such as graphical user interfaces (GUIs), visualization capabilities, and the use of many sensors of different types. Therefore, many software engineering issues such as reusability and the use of commercial off-the-shelf (COTS) components [35.23], real-time issues [35.24–26], sensor selection [35.27], reliability [35.28–30], and embedded testing [35.31] are now receiving more attention from system developers.

Each sensor type has different characteristics and functional descriptions. Consequently, some approaches aim to develop general methods of modeling sensor systems in a manner that is independent of the physical sensors used. In turn, this enables the performance and robustness of multisensor systems to be studied in a general way. There have been many attempts to provide *the* general model, along with its mathematical basis and description. Some of these modeling techniques concern error analysis and fault tolerance of multisensor systems [35.32–37]. Other techniques are model based, and require a priori knowledge of the sensed object and its environment [35.38–40]. These help fit data to a model, but do not always provide the means to compare alternatives. Task-directed sensing is another approach to devising sensing strategies [35.41–43]. General sensor modeling work has had a considerable influence on the evolution of multisensor fusion architectures.

Another approach to modeling sensor systems is to define sensori-computational systems associated with each sensor to allow design, comparison, transformation, and reduction of any sensory system [35.44]. In this approach, the concept of an information invariant is used to define a measure of information complexity. This provides a computational theory allowing analysis, comparison, and reduction of sensor systems.

In general terms, multisensor fusion architectures may be classified according to the choice along four independent design dimensions:

1. Centralized–decentralized,
2. Local–global interaction of components,
3. Modular–monolithic, and
4. Heterarchical–hierarchical.

The most prevalent combinations are:

- Centralized, global interaction, and hierarchical,
- Decentralized, global interaction, and heterarchical,
- Decentralized, local interaction, and hierarchical,
- Decentralized, local interaction, and heterarchical.

In some cases explicit modularity is also desirable. Most existing multisensor architectures fit reasonably well into one of these categories. These categories make no general commitment to the algorithmic architecture.

Avidbots Ex1016
Page 915 of 2154

If the algorithmic architecture is the predominant feature of a system, it will be characterized as part of multisensor fusion theory in Sect. 35.1; otherwise, it merely differentiates methods within one of the four meta-architectures.

35.2.2 Centralized, Local Interaction, and Hierarchical

Centralized, local interaction and hierarchical architectures encompass a number of system philosophies. Least representationally demanding is the *subsumption architecture* initially proposed by *Braitenberg* [35.45] and popularized by *Brooks* [35.46]. The subsumption multisensor architecture defines behaviors as the basic components, and employs a layered set of behaviors to embody one program (monolithic). Any behavior may utilize the output of other behaviors, and may also inhibit other behaviors. The hierarchy is defined by the layers, although this is not always clear-cut. The major design philosophy is to develop behaviors directly from perception–action loops without recourse to brittle, environment representations. This leads to robustness in operation, but a lack of composite behavior predictability.

A more sophisticated (representationally) behavior-based system is the *distributed field robot architecture* (DFRA) [35.47]. This is a generalization of the sensor fusion effects (SFX) architecture [35.48]. This approach exploits modularity, and aims to achieve both behavior-based and deliberative action, reconfigurability and interoperability through the use of Java, Jini, and XML, fault tolerance, adaptability, longevity, consistent interfaces, and dynamic components. The algorithmic architecture is based on fuzzy logic controllers. Experiments have been demonstrated on outdoor mobile robot navigation.

Other similar architectures of this type include perception action networks *Lee* and *Ro* [35.49, 50], while *Draper* et al. [35.51] focuses on types of information needed to perform tasks (higher-level integration); see also [35.52].

Another approach to this type of sensor fusion is to use artificial neural networks. The advantage is that the user, at least in principle, does not need to understand how sensor modalities relate, nor model the uncertainties, nor in fact determine the structure of the system more than to specify the number of layers in the network and the number of nodes per layer. The neural network is presented with a set of training examples, and must determine through the weights on the neuron connections the optimal mapping from inputs to desired outputs (classifications, control signals, etc.) [35.53, 54].

Various other methods exist; for example, *Hager* and *Mintz* [35.42, 43] defines a task-oriented approach to sensor fusion based on Bayesian decision theory and develops an object-oriented programming framework. *Joshi* and *Sanderson* [35.55] describe a

> *methodology for addressing model selection and multisensor fusion issues using representation size (description length) to choose (1) model class and number of parameters, (2) model parameter resolution (3) subset of observed features to model, and (4) correspondence to map features to models.*

Their approach is broader than an architecture and uses a minimization criterion to synthesize a multisensor fusion system to solve specific two-dimensional (2-D) and three-dimensional (3-D) object recognition problems.

35.2.3 Decentralized, Global Interaction, and Heterarchical

The major example of the decentralized, global interaction meta-architecture is the blackboard system. There have been many examples of blackboard systems developed for data fusion applications. For example, the SEPIA system of *Berge-Cherfaoui* and *Vachon* [35.56] uses logical sensors (see below) in the form of modular agents that post results to a blackboard. The overall architectural goals for blackboards include efficient collaboration and dynamic configuration. Experiments are reported on an indoor robot moving from room to room.

The MESSIE (multi expert system for scene interpretation and evaluation) system [35.57] is a scene interpretation system based on multisensor fusion; it has been applied to the interpretation of remotely sensed images. A typology of the multisensor fusion concepts is presented, and the consequences of modeling problems for objects, scene, and strategy are derived. The proposed multispecialist architecture generalized the ideas of their previous work by taking into account the knowledge of sensors, the multiple viewing notion (shot), and the uncertainty and imprecision of models and data modeled with possibility theory. In particular, generic models of objects are represented by concepts independent of sensors (geometry, materials, and spatial context). Three kinds of specialists are present in the architecture: generic specialists (scene and conflict), semantic object specialists, and low-level specialists. A blackboard structure with a centralized control is used. The interpreted scene is implemented as a matrix of pointers enabling conflicts to be detected very easily. Under the control of the scene specialist, the conflict

Avidbots Ex1016
Page 916 of 2154

specialist resolves conflicts using the spatial context knowledge of objects. Finally, an interpretation system with SAR (synthetic aperture radar)/SPOT sensors is described, and an example of a session concerned with bridge, urban area, and road detection is shown.

35.2.4 Decentralized, Local Interaction, and Hierarchical

One of the earliest proposals for this type of architecture is the real-time control system (RCS) [35.58]. RCS is presented as a cognitive architecture for intelligent control, but essentially uses multisensor fusion to achieve complex control. RCS focuses on task decomposition as the fundamental organizing principle. It defines a set of nodes, each comprised of a sensor processor, a world model, and a behavior generation component. Nodes communicate with other nodes, generally in a hierarchical manner, although across-layer connections are allowed. The system supports a wide variety of algorithmic architectures, from reactive behavior to semantic networks. Moreover, it maintains signals, images, and maps, and allows tight coupling between iconic and symbolic representations. The architecture does not generally allow dynamic reconfiguration, but maintains the static module connectivity structure of the specification. RCS has been demonstrated in unmanned ground vehicles [35.59]. Other object-oriented approaches have been reported [35.34, 60].

An early architectural approach which advocated strong programming semantics for multisensor systems is the logical sensor system (LSS). This approach exploits functional (or applicative) language theory to achieve that.

The most developed version of LSS is instrumented LSS (ILLS) [35.22]. The ILLS approach is based on the LSS introduced by *Shilcrat* and *Henderson* [35.61]. The LSS methodology is designed to specify any sensor in a way that hides its physical nature. The main goal behind LSS was to develop a coherent and efficient presentation of the information provided by many sensors of different types. This representation provides a means for recovery from sensor failure, and also facilitates reconfiguration of the sensor system when adding or replacing sensors [35.62].

ILSS is defined as an extension to LSS, and is comprised of the following components (Fig. 35.4):

1. *ILS* name: uniquely identifies a module
2. *Characteristic output vector (COV)*: strongly typed output structure, with one output vector and zero or more input vectors
3. *Commands*: input commands to the module, and output commands to the other modules
4. *Select function*: a selector that detects the failure of an alternate and switches to another alternate if possible
5. *Alternate subnets*: alternative ways of producing the COV_{out}; it is these implementations of one or more algorithms that carry the main functions of the module
6. *Control command interpreter (CCI)*: interpreter of the commands to the module
7. *Embedded tests*: self-testing routines that increase robustness and facilitate debugging
8. *Monitors*: modules that check the validity of the resulting COVs
9. *Taps*: hooks on the output lines to view different COV values.

These components identify the system behavior and provide mechanisms for online monitoring and debugging. In addition, they provide handles for measuring the runtime performance of the system. Monitors are validity check stations that filter the output and alert the user to any undesired results. Each monitor is equipped with a set of rules (or constraints) that governs the behavior of the COV under different conditions.

Embedded testing is used for online checking and debugging purposes. *Weller* et al. proposed a sensor-processing model with the ability to detect measurement errors and to recover from these errors [35.31]. This method is based on providing each system module with verification tests to verify certain characteristics in the measured data, and to verify the internal and output data resulting from the sensor-module algorithm. The recovery strategy is based on rules that are local to the

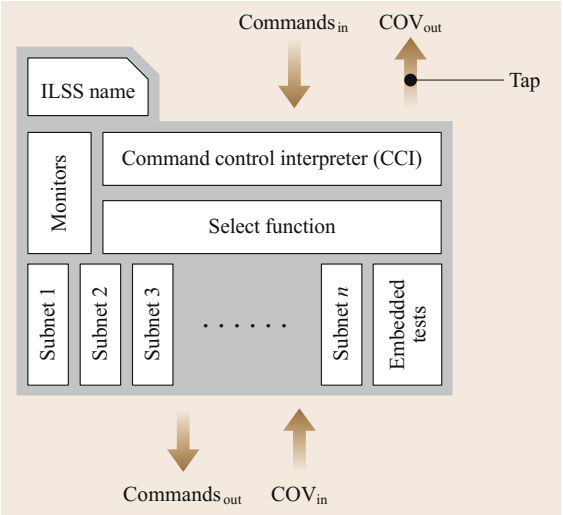


Fig. 35.4 Instrumented logical sensor module

Avidbots Ex1016
Page 917 of 2154

Table 35.1 Canonical components and the roles they play. Multiple ✓ in the same row indicate that some interrole relationships are internalized within a component. *Frame* does not participate in information fusion or decision making but is required for localization and other platform-specific tasks (after [35.21])

Component	Belief			Plan			Action	
Type	Source	Fuse/dist	Sink	Source	Fuse/dist	Sink	Source	Sink
Sensor	✓							
Node		✓			✓			
Actuator								✓
Planner			✓	✓		✓	✓	
UI	✓		✓			✓	✓	
Frame								

different sensor modules. ILSS uses a similar approach called *local embedded testing*, in which each module is equipped with a set of tests based on the semantic definition of that module. These tests generate input data to check different aspects of the module, then examine the output of the module using a set of constraints and rules defined by the semantics. These tests can also take input from other modules to check the operation of a group of modules. Examples are given of a wall-posed estimation system comprised of a Labmate platform with a camera and sonars. Many extensions have been proposed for LSS [35.63, 64].

35.2.5 Decentralized, Local Interaction, and Heterarchical

The best example of this meta-architecture is the active sensor network (ASN) framework for distributed data fusion developed by *Makarenko* et al. [35.21, 65]. The distinguishing features of the various architectures are now described.

Meta-architecture

The distinguishing features of ASN are its commitment to decentralization, modularity, and strictly local interactions (this may be physical or by type). Thus, these are communicating processes. Decentralized means that no component is central to operation of the system, and the communication is peer to peer. Also, there are no central facilities or services (e.g., for communication, name and service lookup or timing). These features lead to a system that is scalable, fault tolerant, and reconfigurable.

Local interactions mean that the number of communication links does not change with the network size. Moreover, the number of messages should also remain constant. This makes the system scalable as well as reconfigurable.

Modularity leads to interoperability derived from interface protocols, reconfigurability, and fault tolerance: failure may be confined to individual modules.

Algorithmic Architecture

There are three main algorithmic components: belief fusion, utility fusion, and policy selection. Belief fusion is achieved by communicating all beliefs to neighboring platforms. A belief is defined as a probability distribution of the world state space.

Utility fusion is handled by separating the individual platform's partial utility into the team utility of belief quality and local utilities of action and communication. The downside is that the potential coupling between individual actions and messages is ignored because the utilities of action and communication remain local.

The communication and action policies are chosen by maximizing expected values. The selected approach is to achieve point maximization for one particular state and follows the work of *Durrant-Whyte* et al. [35.11, 66].

Conceptual Architecture

The data types of the system include:

1. *Beliefs*: Current world beliefs
2. *Plans*: Future planned world beliefs
3. *Actions*: Future planned actions.

The definition of component roles leads to a natural partition of the system.

The information fusion task is achieved through the definition of four component roles for each data type: *source*, *sink*, *fuser*, and *distributor*. (Note that the data type action does not have fuser or distributor component roles.)

Connections between distributors form the backbone of the ASN framework, and the information exchanged is in the form of their local beliefs. Similar considerations are used to determine component roles for the decision-making and system configuration tasks.

Logical Architecture

A detailed architecture specification is determined from the conceptual architecture. It is comprised of

Avidbots Ex1016
Page 918 of 2154

six canonical component types as described in Table 35.1 [35.21].

Makarenko then describes how to combine the components and interfaces to realize the use cases of the problem domain in ASN.

Execution Architecture

The execution architecture traces the mapping of logical components to runtime elements such as processes and shared libraries. The deployment view shows the mapping of physical components onto the nodes of the physical system. The source code view explains how the software implementing the system is organized. At the architectural level, three items are addressed: execution, deployment, and source code organization.

The experimental implementation of the ASN framework has proven to be flexible enough to accommodate a variety of system topologies, platform, and sensor hardware, and environment representations. Several examples are given with a variety of sensors, processors, and hardware platforms.

35.3 Applications

Multisensor fusion systems have been applied to a wide variety of problems in robotics (see the references for this chapter and VIDEO 132, VIDEO 638 and VIDEO 639), but the two most general areas are *dynamic system control* and *environment modeling*. Although there is some overlap between these, they may generally be characterized as:

- *Dynamic system control*: The problem is to use appropriate models and sensors to control the state of a dynamic system (e.g., industrial robot, mobile robot, autonomous vehicle, surgical robot). Usually such systems involve real-time feedback control loops for steering, acceleration, and behavior selection. In addition to state estimation, uncertainty models are required. Sensors may include force/torque sensors, gyros, global positioning system (GPS), position encoders, cameras, range finders, etc..
- *Environment modeling*: The problem is to use appropriate sensors to construct a model of some aspect of the physical environment. This may be a particular object, e.g., a cup, a physical part, a face, or a larger part of the surroundings, e.g., the interior of a building, part of a city or an extended remote or underground area. Typical sensors include cameras, radar, 3-D range finders, infrared (IR), tactile sensors and touch probes (CMMs), etc. The result is usually expressed as geometry (points, lines, surfaces), features (holes, sinks, corners, etc.), or physical properties. Part of the problem includes the determination of optimal sensor placement.

35.3.1 Dynamic System Control

The EMS-Vision system [35.67] is an outstanding exemplar of this application domain. The goal is to develop a robust and reliable perceptual system for autonomous vehicles. The development goals, as stated by the EMS-Vision team, are:

- COTS components
- Wide variety of objects modeled and incorporated into behaviors
- Inertial sensors for ego-state estimation
- Peripheral/foveal/saccadic vision
- Knowledge and goal driven behavior
- State tracking for objects
- 25 Hz real-time update rate.

The approach has been in development since the 1980s. Figure 35.5 shows the first vehicle to drive fully autonomously on the German autobahn for 20 km and at speeds up to 96 km/h.

Information from inertial and vision sensors is combined to produce a road scene tree (Fig. 35.6). A four-



Fig. 35.5 First fully autonomous vehicle on German autobahn

Avidbots Ex1016
Page 919 of 2154

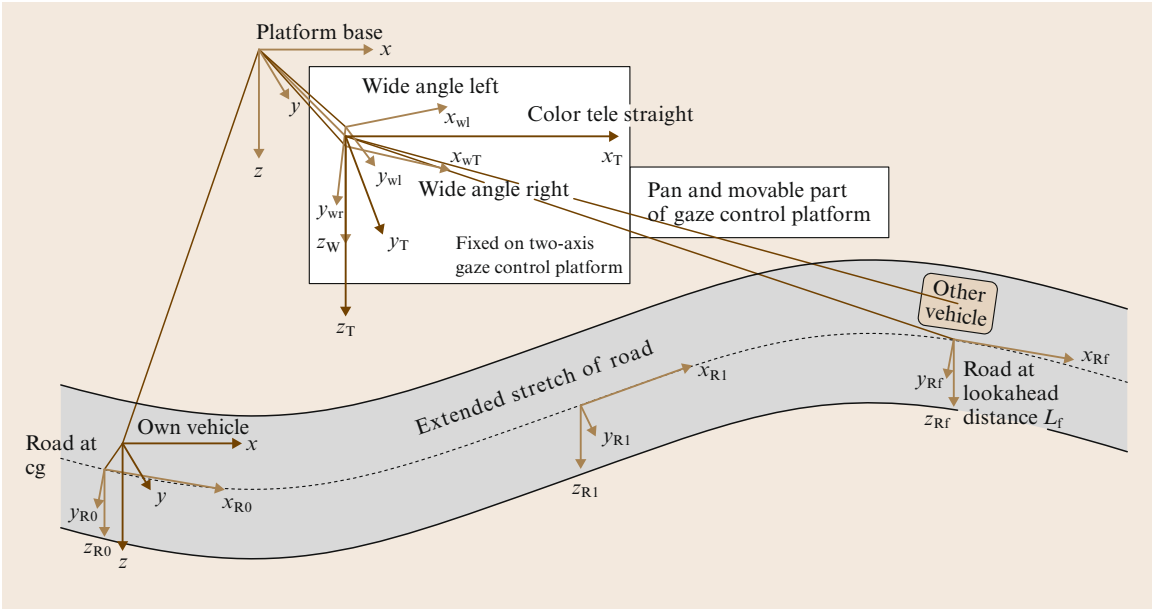


Fig. 35.6 EMS-Vision road scene tree

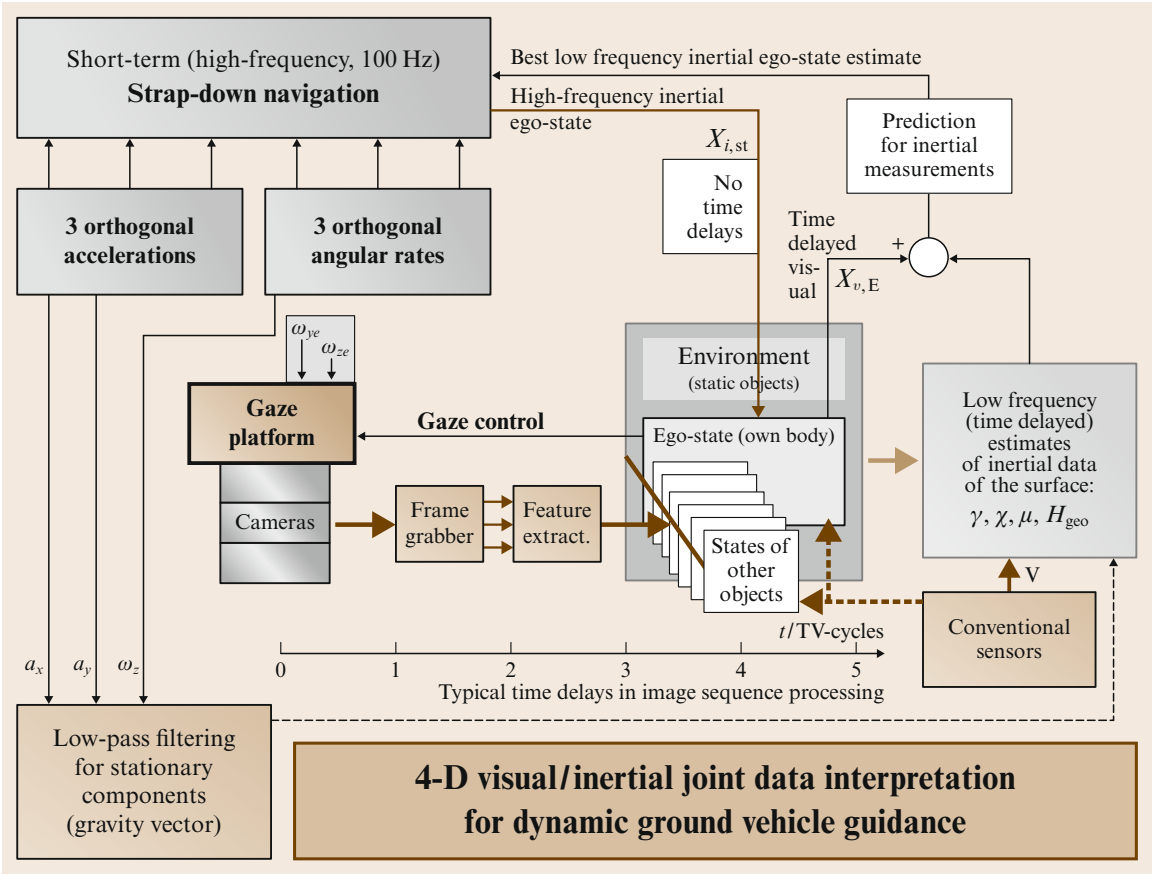


Fig. 35.7 EMS-Vision guidance system

Avidbots Ex1016
Page 920 of 2154

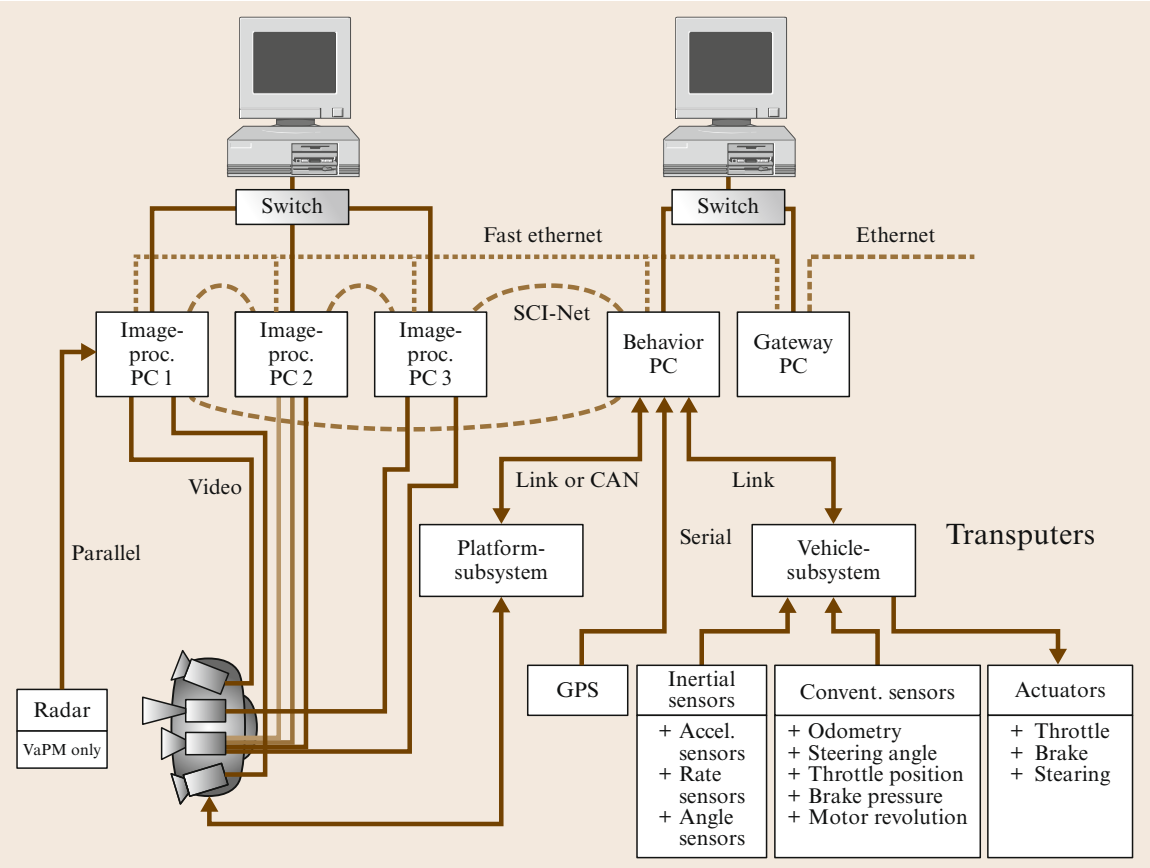


Fig. 35.8 EMS-Vision hardware layout (CAN: controller area network)

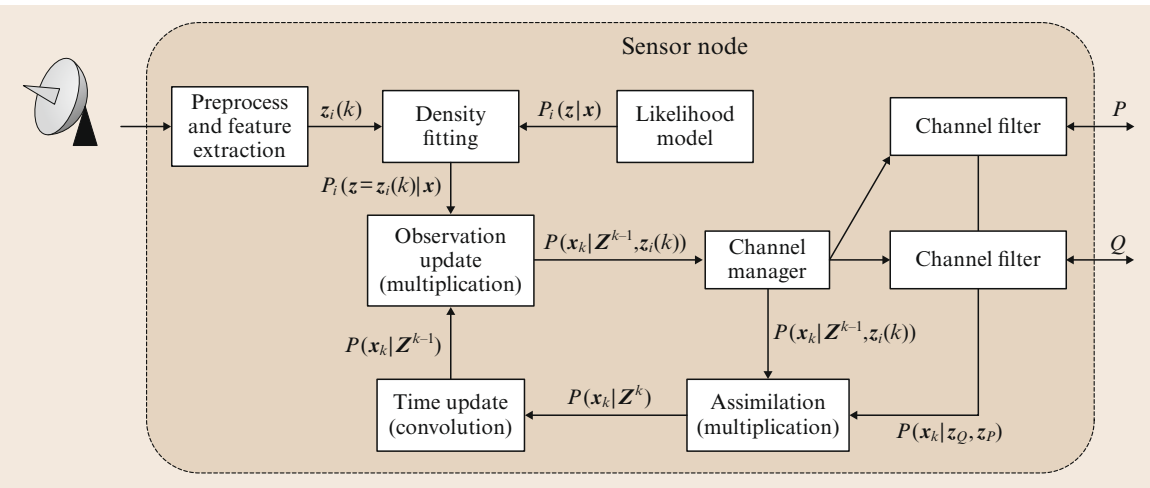


Fig. 35.9 Mathematical structure of a decentralized data fusion node

Avidbots Ex1016
Page 921 of 2154

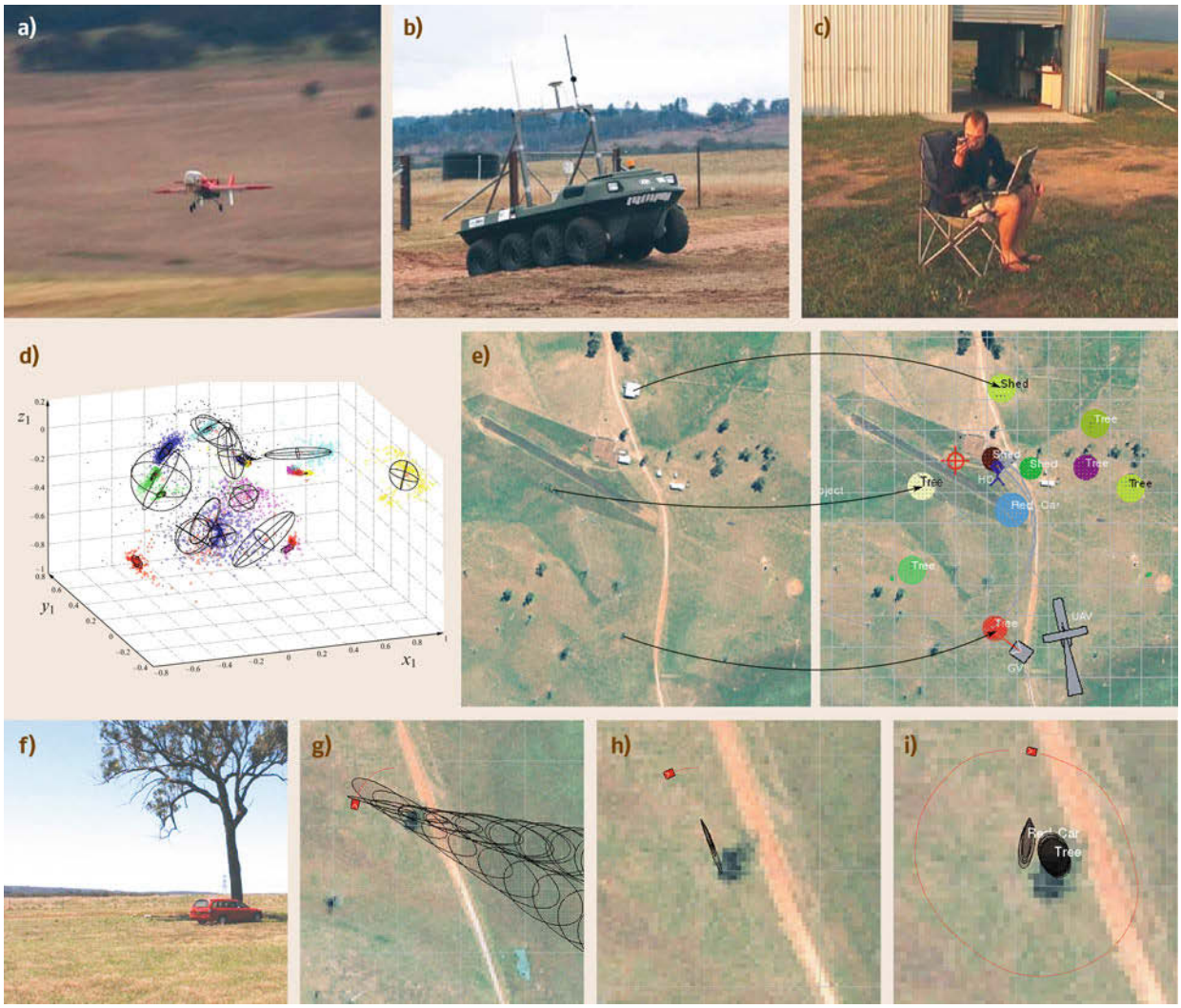


Fig.35.10a–i A synopsis of the ANSER II autonomous network and its operation. **(a–c)** Main system components: **(a)** air vehicle, **(b)** ground vehicle, **(c)** human operative. **(d–e)** The perception process: **(d)** top three dimensions of features discovered from ground-based visual sensor data along with the derived mixture model describing these feature properties, **(e)** sector of the overall picture obtained from fusing air vehicle (UAV), ground vehicle (GV), and human operator (HO) information. Each set of ellipses corresponds to a particular feature and the labels represent the identity state with highest probability. **(f–i)** Sequential fusion process for two close landmarks: **(f)** a tree and a red car, **(g)** bearing-only visual observations of these landmarks are successively fused, **(h)** to determine location and identity **(i)**. Note the Gaussian mixture model for the bearing measurement likelihood

dimensional (4-D) generic object representation is built which includes background knowledge of the object (e.g., roads), its behavioral capabilities, object state and variances, and shape and aspect parameters. Figure 35.7 shows the 4-D inertial/vision multisensor guidance system, while Fig. 35.8 shows the hardware aspects.

In summary, the EMS-Vision system is an interesting and powerful demonstration of multisensor fusion for dynamic system control.

35.3.2 ANSER II: Decentralized Data Fusion

Decentralized data fusion (DDF) methods were initially motivated by the insight that the *information* or canonical form of the conventional Kalman filter data fusion algorithm could be implemented by simply adding information contributions from observations as shown in (35.41). As these (vector and matrix) additions are commutative, the update or data fusion process can

Avidbots Ex1016
Page 922 of 2154

be optimally distributed amongst a network of sensors [35.11, 12, 68]. The aim of the ANSER II project was to generalize the DDF method to deal with non-Gaussian probabilities for observations and states, and to incorporate information from a diversity of sources including uninhabited air and ground vehicles, terrain databases, and human operatives.

The mathematical structure of a DDF sensor node is shown in Fig. 35.9. The sensor is modeled directly in the form of a likelihood function. Once instantiated with an observation, the likelihood function is input to a local fusion loop which implements a local form of the Bayesian time and observation update of (35.7) and (35.8). Network nodes accumulate probabilistic information from observation or communication and exchange mutual information (information gain) with other nodes in the network [35.21]. This mutual information is transmitted to and assimilated by other nodes in the network in an ad hoc manner. The result is that all nodes in the network obtain a single integrated posterior-probability-based all-node observations.

The ANSER II system consists of a pair of autonomous air vehicles equipped with infrared and visual sensors, a pair of unmanned ground vehicles equipped with visual and radar sensors, and additional information provided by geometric and hyperspectral databases, along with information input by human operatives [35.69]. The likelihood functions for single-sensor features are obtained through a semisupervised machine learning method [35.70]. The resulting probabilities are modeled in the form of a mixture of Gaussians. Each platform then maintains a bank of decentralized, non-Gaussian Bayesian filters for the observed features, and transmits this information to all other platforms. The net result is that each platform maintains a complete map of all features observed by all nodes in the network. Multiple observations of the same feature, possibly by different platforms, result in an increasingly accurate estimate of the feature location for all nodes. A corresponding discrete probability measure is used for Fig. 35.10 shows a synopsis of the operation of the ANSER II system.

The ANSER II system demonstrates a number of general principles in Bayesian data fusion methods, specifically the need to model sensors appropriately through the likelihood function, and the possibility of building very different data fusion architectures from the essential Bayesian form.

35.3.3 Recent Developments

There have been recent developments in multisensor fusion methods: see [35.71–73] for more theoretical work, vision and biomedical applications [35.74–77], tracking and terrain classification [35.78–81], and robotics and automation [35.82–87].

35.4 Conclusions

Multisensor data fusion has progressed greatly in the last few decades; further advances in the field will be documented in the robotics and multisensor fusion and integration conference and journal literature. Robust applications are being fielded based on the body of theory and experimental knowledge produced by the research community. Current directions of interest include:

1. Large-scale, ubiquitous sensor systems,
2. Bio-based or biomimetic systems,
3. Medical in situ applications,
4. Wireless sensor networks.

Representative large-scale examples include intelligent vehicle and road systems, as well as instrumented contexts such as cities. Biological principles may provide fundamentally distinct approaches to the exploitation of dense, redundant, correlated, noisy sensors, especially when considered as part of a Gibbsian framework for behavioral response to environmental stimuli. Another issue here is the development of a theoretical understanding of sensor system development, adaptivity, and learning with respect to the particular context in which the system is deployed.

Further pushing the envelope of both technology and theory will permit the introduction of micro- and nanosensors into the human body and allow the monitoring and locally adaptive treatment of various illnesses. Finally, a more-complete theoretical framework that encompasses system models for wireless sensor networks is still required. This should include models of the physical phenomena being monitored, as well as operational and network issues. Finally, numerical analysis of the algorithmic properties of data-driven systems with sensor data error sources must be unified with the analysis of truncation, roundoff, and other errors.

A firm foundation exists upon which to build these new theories, systems, and applications. It will be a vibrant area of research for years to come.

Avidbots Ex1016
Page 923 of 2154

Video-References

VIDEO 132 AnnieWay
available from http://handbookofrobotics.org/view-chapter/35/videodetails/132
VIDEO 638 Application of visual odometry for sewer inspection robots
available from http://handbookofrobotics.org/view-chapter/35/videodetails/638
VIDEO 639 Multisensor remote surface inspection
available from http://handbookofrobotics.org/view-chapter/35/videodetails/639

References

35.1 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
35.2 J.O. Berger: *Statistical Decision Theory and Bayesian Analysis* (Springer, Berlin, Heidelberg 1985)
35.3 A. Elfes: Sonar-based real-world mapping and navigation, IEEE Trans. Robotics Autom. **3**(3), 249–265 (1987)
35.4 L. Matthies, A. Elfes: Integration of sonar and stereo range data using a grid-based representation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1988) pp. 727–733
35.5 L.D. Stone, C.A. Barlow, T.L. Corwin: *Bayesian Multiple Target Tracking* (Artech House, Norwood 1999)
35.6 Y. Bar-Shalom: *Multi-Target Multi-Sensor Tracking* (Artec House, Norwood 1990)
35.7 Y. Bar-Shalom, T.E. Fortmann: *Tracking and Data Association* (Academic, New York 1988)
35.8 P.S. Maybeck: *Stochastic Models, Estimaton and Control* (Academic, New York 1979)
35.9 W. Sorensen: Special issue on the applications of the Kalman filter, IEEE Trans. Autom. Control **28**(3), 254–255 (1983)
35.10 B.D.O. Anderson, J.B. Moore: *Optimal Filtering* (Prentice Hall, Englewood Cliffs 1979)
35.11 J. Manyika, H.F. Durrant-Whyte: *Data Fusion and Sensor Management: An Information-Theoretic Approach* (Ellis Horwood, New York 1994)
35.12 S. Sukkarieh, E. Nettleton, J.H. Kim, M. Ridley, A. Goktogan, H. Durrant-Whyte: The ANSER project: Data fusion across multiple uninhabited air vehicles, Int. J. Robotics Res. **22**(7), 505–539 (2003)
35.13 R.E. Moore: *Interval Analysis* (Prentice Hall, Englewood Cliffs 1966)
35.14 D. Dubois, H. Prade: *Fuzzy Sets and Systems: Theory and Applications* (Academic, New York 1980)
35.15 S. Blackman, R. Popoli: *Design and Analysis of Modern Tracking Systems* (Artec House, Boston 1999)
35.16 D. Pagac, E.M. Nebot, H. Durrant-Whyte: An evidential approach to map-building for autonomous vehicles, IEEE Trans. Robotics Autom. **14**(4), 623–629 (1998)
35.17 D. Hall, J. Llinas: *Handbook of Multisensor Data Fusion* (CRC, Boca Raton 2001)
35.18 E.L. Waltz, J. Llinas: *Sensor Fusion* (Artec House, Boston 1991)
35.19 M. Kam, Z. Zhu, P. Kalata: Sensor fusion for mobile robot navigation, Proceedings IEEE **85**, 108–119 (1997)
35.20 H. Carvalho, W. Heinzelman, A. Murphy, C. Coelho: A general data fusion architecture, Proc. 6th Int. Conf. Inf. Fusion, Cairns (2003)
35.21 A. Makarenko: A Decentralized Architecture for Active Sensor Networks, Ph.D. Thesis (University of Sydney, Sydney 2004)
35.22 M. Dekhil, T. Henderson: Instrumented logical sensors systems, Int. J. Robotics Res. **17**(4), 402–417 (1998)
35.23 J.A. Profeta: Safety-critical systems built with COTS, Computer **29**(11), 54–60 (1996)
35.24 H. Hu, J.M. Brady, F. Du, P. Probert: Distributed real-time control of a mobile robot, J. Intell. Autom. Soft Comput. **1**(1), 63–83 (1995)
35.25 S.A. Schneider, V. Chen, G. Pardo: ControlShell: A real-time software framework, AIAA Conf. Intell. Robotics Field Fact. Serv. Space (1994)
35.26 D. Simon, B. Espiau, E. Castillo, K. Kapellos: Computer-aided design of a generic robot controller handling reactivity and real-time issues, IEEE Trans. Control Syst. Technol. **4**(1), 213–229 (1993)
35.27 C. Giraud, B. Jouvencel: Sensor selection in a fusion process: a fuzzy approach, Proc. IEEE Int. Conf. Multisens. Fusion Integr., Las Vegas (1994) pp. 599–606
35.28 R. Kapur, T.W. Williams, E.F. Miller: System testing and reliability techniques for avoiding failure, Computer **29**(11), 28–30 (1996)
35.29 K.H. Kim, C. Subbaraman: Fault-tolerant real-time objects, Communication ACM **40**(1), 75–82 (1997)
35.30 D.B. Stewart, P.K. Khosla: Mechanisms for detecting and handling timing errors, Communication ACM **40**(1), 87–93 (1997)
35.31 G. Weller, F. Groen, L. Hertzberger: A sensor processing model incorporating error detection and recovery. In: *Traditional and Non-Traditional Robotic Sensors*, ed. by T. Henderson (Springer, Berlin, Heidelberg 1990) pp. 351–363
35.32 R.R. Brooks, S. Iyengar: *Averaging Algorithm for Multi-Dimensional Redundant Sensor Arrays: Resolving Sensor Inconsistencies*, Tech. Rep. (Louisiana State University, Baton Rouge 1993)
35.33 T.C. Henderson, M. Dekhil: *Visual Target Based Wall Pose Estimation*, Tech. Rep. UUCS-97-010 (University of Utah, Salt Lake City 1997)

Avidbots Ex1016
Page 924 of 2154

35.34 S. Iyengar, D. Jayasimha, D. Nadig: A versatile architecture for the distributed sensor integration problem, Computer **43**, 175–185 (1994)
35.35 D. Nadig, S. Iyengar, D. Jayasimha: A new architecture for distributed sensor integration, Proc. IEEE Southeastcon (1993)
35.36 L. Prasad, S. Iyengar, R.L. Kashyap, R.N. Madan: Functional characterization of fault tolerant integration in distributed sensor networks, IEEE Trans. Syst. Man Cybern. **25**, 1082–1087 (1991)
35.37 L. Prasad, S. Iyengar, R. Rao, R. Kashyap: Fault-tolerence sensor integration using multiresolution decomposition, Am. Phys. Soc. **49**(4), 3452–3461 (1994)
35.38 H.F. Durrant-Whyte: *Integration, Coordination, and Control of Multi-Sensor Robot Systems* (Kluwer, Boston 1987)
35.39 F. Groen, P. Antonissen, G. Weller: Model based robot vision, IEEE Instrum. Meas. Technol. Conf. (1993) pp. 584–588
35.40 R. Joshi, A.C. Sanderson: Model-based multisensor data fusion: A minimal representation approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994)
35.41 A.J. Briggs, B.R. Donald: Automatic sensor configuration for task-directed planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 1345–1350
35.42 G. Hager: *Task Directed Sensor Fusion and Planning* (Kluwer, Boston 1990)
35.43 G. Hager, M. Mintz: Computational methods for task-directed sensor data fusion and sensor planning, Int. J. Robotics Res. **10**(4), 285–313 (1991)
35.44 B. Donald: On information invariants in robotics, Artif. Intell. **72**, 217–304 (1995)
35.45 V. Braitenberg: *Vehicles: Experiments in Synthetic Psychology* (MIT Press, Cambridge 1984)
35.46 R.A. Brooks: A robust layered control system for a mobile robot, IEEE Trans. Robotics Autom. **2**(1), 14–23 (1986)
35.47 K.P. Valavanis, A.L. Nelson, L. Doitsidis, M. Long, R.R. Murphy: *Validation of a Distributed Field Robot Architecture Integrated with a Matlab Based Control Theoretic Environment: A Case Study of Fuzzy Logic Based Robot Navigation*, CRASAR Tech. Rep. 25 (University of South Florida, Tampa 2004)
35.48 R.R. Murphy: *Introduction to AI Robotics* (MIT Press, Cambridge 2000)
35.49 S. Lee: Sensor fusion and planning with perception-action network, Proc. IEEE Conf. Multisens. Fusion Integr. Intell. Syst., Washington (1996)
35.50 S. Lee, S. Ro: Uncertainty self-management with perception net based geometric data fusion, Proc. IEEE Conf. Robotics Autom. (ICRA), Albuquerque (1997)
35.51 B.A. Draper, A.R. Hanson, S. Buluswar, E.M. Riseman: Information acquisition and fusion in the mobile perception laboratory, Proc. SPIE Sens. Fusion VI (1993)
35.52 S.S. Shafer, A. Stentz, C.E. Thorpe: An architecture for sensor fusion in a mobile robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 2002–2007
35.53 S. Nagata, M. Sekiguchi, K. Asakawa: Mobile robot control by a structured hierarchical neural network, IEEE Control Syst. Mag. **10**(3), 69–76 (1990)
35.54 M. Pachter, P. Chandler: Challenges of autonomous control, IEEE Control Syst. Mag. **18**(4), 92–97 (1998)
35.55 R. Joshi, A.C. Sanderson: *Multisensor Fusion* (World Scientific, Singapore 1999)
35.56 V. Berge-Cherfaoui, B. Vachon: Dynamic configuration of mobile robot perceptual system, Proc. IEEE Conf. Multisens. Fusion Integr. Intell. Syst., Las Vegas (1994)
35.57 V. Clement, G. Giraudon, S. Houzelle, F. Sandakly: *Interpretation of Remotely Sensed Images in a Context of Multisensor Fusion Using a Multi-Specialist Architecture*, Rapp. Rech.: No. 1768 (INRIA, Sophia-Antipolis 1992)
35.58 J. Albus: RCS: A cognitive architecture for intelligent multi-agent systems, Proc. IFAC Symp. Intell. Auton. Veh., Lisbon (2004)
35.59 R. Camden, B. Bodt, S. Schipani, J. Bornstein, R. Phelps, T. Runyon, F. French: *Autonomous Mobility Technology Assessment*, Interim Rep., ARL-MR 565 (Army Research Laboratory, Washington 2003)
35.60 T. Queeney, E. Woods: A generic architecture for real-time multisensor fusion tracking algorithm development and evaluation, Proc. SPIE Sens. Fusion VII, Vol. 2355 (1994) pp. 33–42
35.61 T. Henderson, E. Shilcrat: Logical sensor systems, J. Robotics Syst. **1**(2), 169–193 (1984)
35.62 T. Henderson, C. Hansen, B. Bhanu: The specification of distributed sensing and control, J. Robotics Syst. **2**(4), 387–396 (1985)
35.63 J.D. Elliott: Multisensor Fusion within an Encapsulated Logical Device Architecture, Master's Thesis (University of Waterloo, Waterloo 2001)
35.64 M.D. Naish: Elsa: An Intelligent Multisensor Integration Architecture for Industrial Grading Tasks, Master's Thesis (University of Western Ontario, London 1998)
35.65 A. Makarenko, A. Brooks, S. Williams, H. Durrant-Whyte, B. Grocholsky: A decentralized architecture for active sensor networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 1097–1102
35.66 B. Grocholsky, A. Makarenko, H. Durrant-Whyte: Information-theoretic coordinated control of multiple sensor platforms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Taipei (2003) pp. 1521–1527
35.67 R. Gregor, M. Lützeler, M. Pellkofer, K.-H. Siedersberger, E. Dickmanns: EMS-Vision: A perceptual system for autonomous vehicles, IEEE Trans. Intell. Transp. Syst. **3**(1), 48–59 (2002)
35.68 B. Rao, H. Durrant-Whyte, A. Sheen: A fully decentralized multi-sensor system for tracking and surveillance, Int. J. Robotics Res. **12**(1), 20–44 (1993)
35.69 B. Upcroft: Non-gaussian state estimation in an outdoor decentralised sensor network, Proc. IEEE Conf. Decis. Control (CDC) (2006)
35.70 S. Kumar, F. Ramos, B. Upcroft, H. Durrant-Whyte: A statistical framework for natural feature repre-

Avidbots Ex1016
Page 925 of 2154

sentation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Edmonton (2005) pp. 1–6
35.71 S. Gao, Y. Zhong, W. Li: Random weighting method for multisensor data fusion, IEEE Sens. J. **11**(9), 1955–1961 (2011)
35.72 M. Lhuillier: Incremental fusion of structure-from-motion and GPS using constrained bundle adjustments, IEEE Trans. Pattern Anal. Mach. Intell. **34**(12), 2489–2495 (2012)
35.73 S. Yu, L. Tranchevent, X. Liu, W. Glanzel, J. Suykens, B. DeMoor, Y. Moreau: Optimized data fusion for kernel k-means clustering, IEEE Trans. Pattern Anal. Mach. Intell. **34**(5), 1031–1039 (2012)
35.74 K. Kolev: Fast joint estimation of silhouettes and dense 3D geometry from multiple images, IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 493–505 (2012)
35.75 C. Loy: Incremental activity modeling in multiple disjoint cameras, IEEE Trans. Pattern Anal. Mach. Intell. **34**(9), 1799–1813 (2012)
35.76 N. Poh, J. Kittler: A unified framework for biometric expert fusion incorporating quality measures, IEEE Trans. Pattern Anal. Mach. Intell. **34**(1), 3–18 (2012)
35.77 M.-F. Weng, Y.-Y. Chuang: Cross-domain multicue fusion for concept-based video indexing, IEEE Trans. Pattern Anal. Mach. Intell. **34**(10), 1927–1941 (2012)
35.78 M. Hwangbo, J.-S. Kim, T. Kanade: Gyro-aided feature tracking for a moving camera: fusion, auto-calibration and GPU implementation, Intl. J. Robotics Res. **30**(14), 1755–1774 (2011)
35.79 H. Seraji, N. Serrano: A multisensor decision fusion system for terrain safety assessment, IEEE Trans. Robotics **25**(1), 99–108 (2009)
35.80 H. Himberg, Y. Motai, A. Bradley: Interpolation volume calibration: a multisensor calibration technique for electromagnetic trackers, IEEE Trans. Robotics **28**(5), 1120–1130 (2012)
35.81 K. Zhou, S.I. Roumeliotis: Optimal motion strategies for range-Only constrained multisensor target tracking, IEEE Trans. Robotics **24**(5), 1168–1185 (2008)
35.82 N.R. Ahmed, E.M. Sample, M. Campbell: Bayesian multicategorical soft data fusion for human–robot collaboration, IEEE Trans. Robotics **PP**(99), 1–18 (2012)
35.83 H. Frigui, L. Zhang, P.D. Gader: Context-dependent multisensor fusion and its application to land mine detection, IEEE Trans. Geosci. Remote Sens. **48**(6), 2528–2543 (2010)
35.84 J.G. Garcia, A. Robertson, J.G. Ortega, R. Johansson: Sensor fusion for compliant robot motion control, IEEE Trans. Robotics **24**(2), 430–441 (2008)
35.85 R. Heliot, B. Espiau: Multisensor input for CPG-based sensory–motor coordination, IEEE Trans. Robotics **24**(1), 191–195 (2008)
35.86 S. Liu, R.X. Gao, D. John, J.W. Staudenmayer, P.S. Freedson: Multisensor data fusion for physical activity assessment, IEEE Trans. Bio-Med. Eng. **59**(3), 687–696 (2012)
35.87 A. Martinelli: Vision and IMU data fusion: closed-Form solutions for attitude, speed, absolute scale, and bias determination, IEEE Trans. Robotics **28**(1), 44–60 (2012)

Avidbots Ex1016
Page 926 of 2154

Multimedia Contents



Part D Manipulation and Interfaces

Ed. by Makoto Kaneko

36 Motion for Manipulation Tasks
James Kuffner, Pittsburgh, USA
Jing Xiao, Charlotte, USA

37 Contact Modeling and Manipulation
Imin Kao, Stony Brook, USA
Kevin M. Lynch, Evanston, USA
Joel W. Burdick, Pasadena, USA

38 Grasping
Domenico Prattichizzo, Siena, Italy
Jeffrey C. Trinkle, Troy, USA

39 Cooperative Manipulation
Fabrizio Caccavale, Potenza, Italy
Masaru Uchiyama, Sendai, Japan

40 Mobility and Manipulation
Oliver Brock, Berlin, Germany
Jaeheung Park, Suwon, Korea
Marc Toussaint, Stuttgart, Germany

41 Active Manipulation for Perception
Anna Petrovskaya, Stanford, USA
Kaijen Hsiao, Palo Alto, USA

42 Haptics
Blake Hannaford, Seattle, USA
Allison M. Okamura, Stanford, USA

43 Telerobotics
Günter Niemeyer, Glendale, USA
Carsten Preusche, Wessling, Germany
Stefano Stramigioli, Enschede, The Netherlands
Dongjun Lee, Seoul, Korea

44 Networked Robots
Dezhen Song, College Station, USA
Ken Goldberg, Berkeley, USA
Nak-Young Chong, Ishikawa, Japan

Part D, Manipulation and Interfaces, is separated into two subparts; the first half is concerned with manipulation where frameworks of modeling, motion planning, and control of grasp and manipulation of an object are addressed, and the second half is concerned with interfaces where physical human–robot interactions are handled. Humans can achieve grasping and manipulation of an object dexterously through hand–arm coordination. An optimum control skill for such a redundant system is naturally and gradually acquired through experience in our daily life. Especially, fingers play an important role for expressing human dexterity. Without dexterous fingers, it is hard for us to handle any daily tool, such as a pencil, keyboard, cup, knife, or fork. This dexterity is supported with active and passive compliance as well as the multiple sensory organs existing at the fingertip. Such dexterous manipulation enables us to clearly differentiate humans from other animals. Thus, manipulation is one of the most important functions for humans. We eventually acquired the current shape of finger, the sensory organs, and skill for manipulation, through a long history of evolution, over more than six million years. While humans and robots are largely different in terms of actuators, sensors, and mechanisms, achieving dexterous manipulation like that of a human in a robot is a challenging subject in robotics. As we overview current robot technology, however, we observe that the dexterity of robots is still far behind that of humans. With this overview, we now provide a brief synopsis of each chapter in the first half of Part D.

Chapter 36, Motion for Manipulation Tasks, discusses algorithms that generate motion for manipulation tasks at the arm level, especially in an environment, by using the configuration space formalism. While in previous chapters (6 and 7) the focus was on specific algorithmic techniques for robot motion, this chapter is focused on a specific application for robot manipulation. The important example of assembly motion is discussed through the analysis of contact states and compliant motion control.

Chapter 37, Contact Modeling and Manipulation, provides the contact modeling on rigid contact with and without friction, and also the modeling on soft contact, such as elastic and viscoelastic contact interfaces. Kinematics and mechanics with friction are precisely handled under rigid-body contact. The selection matrix $\mathbf{H}$ is introduced to understand the force and velocity constraints at the contact interface. Pushing manipulation is also addressed by using the concept of the friction limit surface.

Chapter 38, Grasping, discusses based on the closure property, grasping with many examples, supposing multifingered robotic hands. A strong constraint for grasping is the unilateral characteristic,where a fingertip can push but not pull an object through a contact point. The rigid-body model is further extended for considering compliance in grasping with robotic hands having a low number of degrees of freedom (DOFs). Kinematics and closure issues are also addressed under this unilateral constraint.

Chapter 39, Cooperative Manipulators, addresses the strategies for controlling both the motion of cooperative system and the interaction forces between the manipulators and the grasped object when two manipulator arms firmly grasp a common object. It should be noted that this chapter allows the bilateral constraint where both directional force and moment are permissible. A general cooperative task-space formulation is introduced for different classes of multiarm robotic systems.

Chapter 40, Mobile Manipulation, focuses on research conducted on an experimental platform that combines capabilities in mobility and manipulation. Furthermore, it involves the interaction between the robots and real-world environment, unstructured environments. The main research objective here is to maximize task generality of autonomous robotic systems, under minimizing the dependence on task-specific, hard-coded, or narrowly-relevant information.

Chapter 41, Active Manipulation for Perception, covers perceptual methods in which manipulation is an integral part of perception. There are advantages to use manipulation rather than vision for perception. For example, manipulation can be used to sense in poor-visibility conditions, and also to determine properties that require physical interaction. This chapter includes the methods that have been developed for inference, planning, recognition, and modeling in sensing-via-manipulation approaches are cover.

Without dexterity like that of humans, future robots will not be able to work instead of humans in environments where human cannot enter. In this sense, the implementation of dexterity into robots is one of the highlights of future robot design. Chapters 36–41 provide a good hint for enhancing dexterity for robots.

The second half of Part D addresses interfaces where humans control a robot or multiple robots through direct or indirect contact with robot(s).We now

provide a brief synopsis of each chapter in the second half of Part D.

Chapter 42, Haptics, discusses robotics devices that allow human operators to experience the sense of touch in remote or virtual environment. Two classes of force feedback haptic devices are discussed. One is an admittance device that senses the force applied by the operator and constrains the operator's position to match the appropriate deflection of a simulated object or surface; the other is an impedance haptic device that senses the position of the operator and then applies a force vector to the operator according to the computed behavior of the simulated object or surface. Haptic rendering from real time three-dimensional (3-D) image is also introduced in this chapter.

Chapter 43, Telerobotics, starts with a discussion on the classification of three different concepts: direct control where all slave motions are directly controlled by the user via the master interface, shared control where task execution is shared between direct control and local sensory control, and supervisory control where the user and slave are connected loosely with strong local autonomy. Various control issues such as lossy communication with Internet and operation with mobile robots, are also addressed.

Chapter 44, Networked Robots, focuses on the framework of computer networks which offer extensive computing, memory, and other resources that can dramatically improve performance. The chapter covers a wide span from the history of networked robots as it evolves from teleoperation to cloud robotics to how to build a networked robot. The very recent progress on cloud robotics and potential topics for future research are included later in the chapter.

In Haptics (Chap. 42) direct contact between human and robot is made, while in both Telerobotics (Chap. 43) and Networked Robots (Chap. 44), an appropriate distance between the human and robot is kept. One of the main issues is how to maintain appropriate control performance of the system in the presence of humans or a time lag between humans and robots.

Multimedia Contents

36. Motion for Manipulation Tasks

James Kuffner, Jing Xiao

This chapter serves as an introduction to Part D by giving an overview of motion generation and control strategies in the context of robotic manipulation tasks. Automatic control ranging from the abstract, high-level task specification down to fine-grained feedback at the task interface are considered. Some of the important issues include modeling of the interfaces between the robot and the environment at the different time scales of motion and incorporating sensing and feedback. Manipulation planning is introduced as an extension to the basic motion planning problem, which can be modeled as a hybrid system of continuous configuration spaces arising from the act of grasping and moving parts in the environment. The important example of assembly motion is discussed through the analysis of contact states and compliant motion control. Finally, methods aimed at integrating global planning with state feedback control are summarized.

36.1 **Overview** 898
36.2 **Task-Level Control** 900
36.2.1 Operational Space Control 900
36.2.2 Combined Force and Position Control 901
36.2.3 Operational Space Control of Redundant Mechanisms 901
36.2.4 Combining Mobility and Manipulation 902
36.2.5 Combining Multiple Task Behaviors 902
36.3 **Manipulation Planning** 904
36.3.1 Configuration Space Formalism . 904
36.3.2 Example of a Three-DOF Planar Manipulator 907
36.3.3 Inverse Kinematics Considerations 908
36.3.4 Continuum Manipulation 910
36.4 **Assembly Motion** 911
36.4.1 Topological Contact States 912
36.4.2 Passive Compliance 912
36.4.3 Active Compliant Motion 914
36.4.4 Compliant Motion Under Manipulator Constraints .. 917
36.5 **Unifying Feedback Control and Planning** 918
36.5.1 Feedback Motion Planning 918
36.5.2 Augmenting Global Plans with Feedback 920
36.6 **Conclusions and Further Reading** 920
36.6.1 General Contact Models 921
36.6.2 Cooperative Manipulation Control 921
36.6.3 Control of Branching Mechanisms 921
36.6.4 Nonholonomic Mobile Manipulation 921
36.6.5 Learning Models with Uncertainty 921
36.6.6 Grasping and Regrasp Planning 921
36.6.7 Multiple Parts 922
36.6.8 Planning for Multiple Robots 922
36.6.9 Planning for Closed Kinematic Chains 922
36.6.10 Planning with Movable Obstacles 922
36.6.11 Nonprehensile Manipulation 922
36.6.12 Assembly Motion Extensions 922
Video-References 923
References 923

Avidbots Ex1016
Page 930 of 2154

36.1 Overview

Part D of this handbook is concerned with the interfaces that connect robots to their environment. We differentiate three such interfaces, depicted in Fig. 36.1. The first interface, between the robot and the computer, is primarily concerned with the automatic generation of motion for performing a task. The second interface is concerned with the physical interaction between the robot and the environment. This chapter relates both interfaces in the context of manipulation tasks. It is focused on automatic specification, planning, and execution of motion of the robot manipulator or the manipulated object to meet a task goal under the constraints of the physical environment. Chapters 37–39, 40, and 42 address other important issues pertaining to the second interface, such as the physical characteristics of contacts between the robot and the environment, grasping and active manipulation (i. e., the interaction between the robot and the object for manipulation), the interaction between cooperative manipulators, and the interaction between mobility and manipulation. The third interface, described in Chaps. 41, 43, and 44, lies between humans and robots. Key issues that need to be addressed involve displaying appropriate sensor information to humans, or permitting humans to interactively specify the task or motion a robot is supposed to perform.

Manipulation refers to the process of moving or rearranging objects in the environment [36.1]. To perform a *manipulation task*, a robot establishes physical contact with objects in the environment and subsequently moves these objects by exerting forces and moments. The object to be manipulated can be large or small, and may serve various purposes in relation to the manipulation task (e.g., flipping a switch, opening a door, polishing a surface). In the context of automated assembly and industrial manipulation, the object to be manipulated is often referred to as a *part*. The *end-effector* generally refers to the link of the *manipulator* that makes contact and applies forces to the *part*. The *end-effector* is so named as it is often the end link in a serial kinematic chain. However, complex manipulation tasks may require the simultaneous exertion of multiple forces and moments at different contact points as well as the execution of sequences of force applications to several objects.

To understand the challenge of generating the motion required for the execution of a manipulation task, we will consider the classic example of inserting a peg into a hole. The mobile manipulator shown in Fig. 36.2 is commanded to perform the peg insertion *manipulation task*, whose general problem structure arises in a number of contexts such as compliant assembly. For simplicity, we will only consider the *transfer motion* (Sect. 36.3) and assume that the robot has already established a stable grasp (Chap. 38) and is holding the peg. If the clearance between the peg and the hole is very small, the peg may not be able to enter the hole easily, and often some contact occurs. In order to guide the peg successfully into the hole, the robot has to deal with the possible contact states between the peg and the hole (Sect. 36.4) and select an appropriate sequence of control strategies that will result in successful completion of the task.

The peg-and-hole example illustrates that successful robotic manipulation involves dealing with models of object geometry and contact in the presence of uncertainty. There are various kinds of uncertainty, including modeling, actuation, and sensing uncertainty. To plan motion strategies in the presence of uncertainty poses difficult computational challenges. Historically, motion planning for manipulation has been divided into *gross motion planning* and *fine motion planning*. The former involves reasoning about the manipulator motion on the macro scale and considers its overall global movement

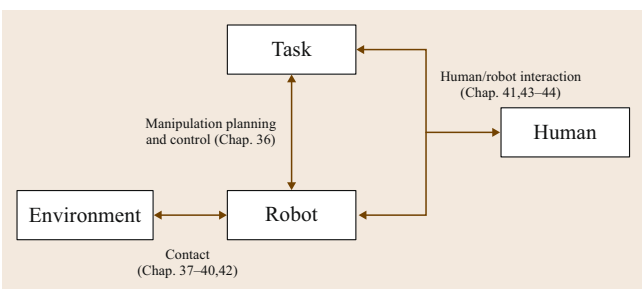


Fig. 36.1 Overview of Part D

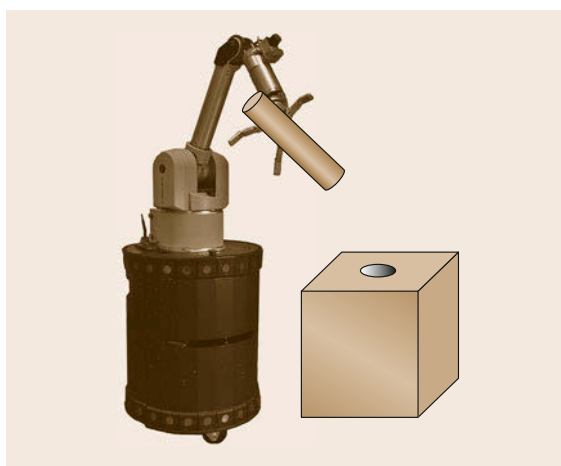


Fig. 36.2 A mobile manipulator inserting a peg into a hole

strategy, while the latter considers how to deal with uncertainty to accomplish a task requiring high precision robustly. The primary reason for the historical division stems from attempts to simplify the various aspects of the problem into something computationally tractable. Assuming that the uncertainty in actuator position, object models, and environment obstacle shapes can be bounded, gross motion planning can be formulated primarily in terms of geometry (Sect. 36.3). As the robot or the manipulated object makes contact with the environment, such as in the example of the insertion task, fine motion planning techniques are employed to take into account contact geometry, forces, friction, and uncertainty effectively (see *Mason*'s book [36.1] and [36.2, 3] for an overview and historical perspective on fine motion strategies for manipulation tasks).

This peg-and-hole insertion example also illustrates that motion in the context of manipulation tasks is subject to *constraints*. These constraints have to be maintained to perform the task successfully. The specific constraints depend on the type of manipulation task, but they may include contact constraints, position, and force constraints for the point at which the manipulator makes contact with the environment, kinematic and dynamic constraints imposed by the mechanism and its actuation capabilities, posture constraints that specify behavior to be performed in addition to the manipulation task, reactive obstacle avoidance in an unpredictably changing environment, and global motion constraints to ensure that a specific goal location is attained. These motion constraints are imposed by the task, by the kinematics of the mechanism performing the task, by the actuation capabilities of the mechanism, and by the environment.

To ensure that constraints are satisfied in spite of uncertainty, sensory information (see Part C – Sensing and Perception) is considered in the context of feedback control loops (Chaps. 8 and 47). Depending on the type of constraint, this feedback has to be considered at various time scales. For example, the exertion of a constant force on an object in the environment requires high-frequency feedback at rates up to 1000 Hz. At the other end of the time scale, we consider changes to the global connectivity of the environment. These changes, resulting from opening doors or moving obstacles, for example, occur relatively slowly or infrequently and feedback about these changes only has to be considered a few times per second. Figure 36.3 graphically illustrates a task-dependent ordering of motion constraints encountered in manipulation tasks and their associated feedback requirements.

This chapter is concerned with algorithms that generate motion for manipulation tasks. Previous chapters already discussed planning algorithms (Chap. 7 and control methods (Chap. 8). While in these two former chapters the focus was on specific algorithmic techniques for robot motion, the current chapter is focused on a specific application, namely robotic manipulation. This application dictates motion constraints and their feedback requirements that will have to be satisfied during the motion for a manipulation task. In contrast, the methods discussed in the two aforementioned chapters each only address a subset of the motion constraints, as indicated in Fig. 36.3.

This chapter first discusses task-level control in Sect. 36.2. Task-level control describes a set of techniques to control a robotic mechanism in terms of its contact points with the environment. These contact points are controlled to perform motion that accomplishes a specific manipulation task. This means that, instead of directly controlling the mechanism, task-relevant points on the mechanism, so-called *operational points*, are controlled. This indirect control provides an intuitive means to specify desired motion for manipulation tasks.

Section 36.3 gives an overview of the configuration space formalization for manipulation planning, and how it can be cast as a classical motion planning problem (Chap. 7). Through this formalism, we can gain an understanding and intuition regarding how the geometry and relative positions of the manipulator, the manipulated objects, and the obstacles in the workspace determine both the geometry and topological structure of the manipulation configuration space. As we shall see, there are often an infinite number of possible ways to accomplish a manipulation task. Thus, the challenge in manipulation planning is to develop methods that deal effectively with the combinatorics of the search over all possible motions.

Section 36.4 addresses planning motion for assembly tasks, for which the peg-in-hole insertion is a typical example. The focus is on motion constrained by contact, called *compliant motion*. There are two broad

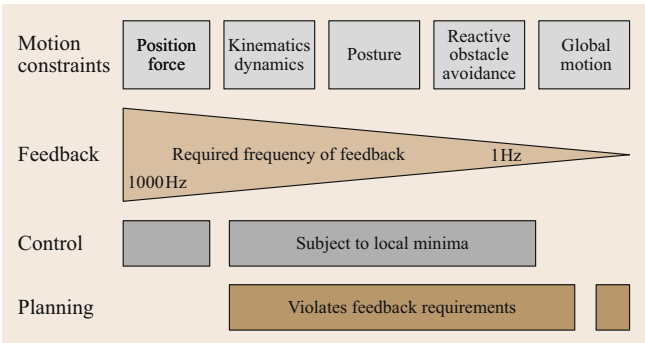


Fig. 36.3 Examples of motion constraints encountered in manipulation tasks and their feedback requirements

classes of assembly strategies involving compliant motion; one class is through special mechanism or control, called *passive compliance*, and the other is through active reasoning of contact states, called *active compliant motion*. Fine motion planning is discussed within the class of active compliant motion.

Methods from task-level control and manipulation planning address aspects of motion in a complementary fashion. While control methods are able to satisfy the feedback requirements of manipulation tasks, they often do not account for the overall global progress in accomplishing the manipulation task. Manipulation planners, on the other hand, reason about the global nature of the task at the high level, but are often computationally too expensive to deal with uncertainty or satisfy the high-frequency feedback requirements of manipulation tasks. A robust and skillful manipulator must satisfy all feedback requirements while at the same time ensuring the accomplishment of the manipulation task. Section 36.5 discusses various techniques that offer a unified view of planning and control for manipulation tasks.

36.2 Task-Level Control

To perform a manipulation task, a robot has to establish contact with the environment. Through contact points the robot is able to apply forces and moments to objects. By controlling the position and velocity of the contact points as well as the forces acting at them, the robot causes the desired motion of objects, thereby performing the manipulation task. The programming of such a task is most conveniently accomplished by directly specifying positions, velocities, and forces at the contact points, rather than by specifying the joint positions and velocities required to achieve them.

Consider the manipulation task of serving a cup of water shown in Fig. 36.2. This task can easily be specified by providing a trajectory for the cup's motion. To determine a joint space trajectory that achieves the same motion, however, would be much more complex. One would have to rely on inverse kinematics (Sect. 2.9), which can be computationally challenging. More importantly, the task constraints imposed on the cup's motion do not uniquely specify the cup's trajectory. For example, while the cup is not allowed to tilt, it can be delivered to the goal location in any vertical orientation. Such task constraints can easily be specified in terms of the object motion – at the task level – but would be very difficult to characterize in terms of joint trajectories. Operational space control is therefore a natural choice for performing manipulation tasks. We will see in Sect. 36.2.3 that the operational space framework provides important additional advantages over joint space control in the context of redundant manipulators.

The advantages associated with task-level control come at the cost of depending on the manipulator Jacobian. For singular configurations of the manipulator (Chap. 4), the manipulator Jacobian is not well defined. In these configurations, task-level control of a manipulator becomes unstable. Special care has to be applied to prevent the manipulator from entering these configurations, or to resort to specialized controllers in the proximity of these configurations. Most commonly, the manipulator is controlled so as to avoid singular configurations.

36.2.1 Operational Space Control

Task-level control – also called operational space control [36.4, 5] (Sect. 8.2) – specifies the behavior of a robot in terms of operational points rather than joint positions and velocities. An operational point is an arbitrary point on the robot that is required to perform a particular motion or to exert a specified force to accomplish a manipulation task. In a serial-chain manipulator arm the operational point is most commonly chosen to coincide with the end-effector. More complex, branching mechanisms, such as humanoid robots, but also redundant serial-chain mechanisms, can have several operational points. To specify positions, velocities, accelerations, forces, and moments at the operational point, it is convenient to define a coordinate system with an origin that coincides with the operation point. This coordinate frame is called the operational frame. The orientation of the frame should be chosen in accordance with the task. For now we will ignore the orientation, however, and only consider tasks that require positioning of the operational point. In Sect. 36.2.2, we will consider more general tasks, combining position and force control.

Let us consider an operational point $\boldsymbol{x}$ on a robot. The relationship between the joint velocities $\dot{\boldsymbol{q}}$ and the velocity of the operational point $\dot{\boldsymbol{x}}$ is given by the Jacobian matrix of the mechanism (Sect. 2.8) at the operational point $\boldsymbol{x}$

$$\dot{\boldsymbol{x}} = \mathbf{J}_x(\boldsymbol{q})\dot{\boldsymbol{q}} \,. \tag{36.1}$$

Note that the Jacobian $\mathbf{J}$ depends on the placement of the operational point $\boldsymbol{x}$, and on the current configura-

tion $\boldsymbol{q}$ of the robot. For simplicity, we will omit this dependency from our notation and just write $\dot{\boldsymbol{x}} = \mathbf{J}\dot{\boldsymbol{q}}$. From (36.1) we can derive an expression for the instantaneous torques $\boldsymbol{\tau}$ acting at the joints when the forces and moments described by $\boldsymbol{F}$ is applied at the operational point $\boldsymbol{x}$

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{F}_{\boldsymbol{x}} \,. \tag{36.2}$$

The vector $\boldsymbol{F}$ captures the task to be performed at operational point $\boldsymbol{x}$. If the task specifies the complete position and orientation of the operational frame, $\boldsymbol{F}$ will be given by $\boldsymbol{F} = (f_x, f_y, f_z, u_x, u_y, u_z)$, where f and u designate forces along and moments about the respective axes. (For easy of presentation, we will refer to $\boldsymbol{F}$ as a force vector, even if it describes forces and moments. The execution of such a task requires a manipulator with at least six degrees of freedom. If a task does not specify one of the components of $\boldsymbol{F}$, this component can be omitted and the corresponding column of the Jacobian matrix is dropped (or a manipulator with fewer than six degrees of freedom can be used to accomplish the task). For example, dropping u_z would indicate that the orientation about the z-axis is not specified by the task. The motion behavior in the dimensions of the task space that have been dropped is unspecified and the robot will float in these dimensions [36.4]. In Sect. 36.2.3 we will discuss how additional behavior can be performed in these unspecified dimensions of the task space.

Similarly to joint space control, we have to account for manipulator dynamics to achieve acceptable performance in operational space control. Using (36.1), we can project the joint space dynamics into operational space to obtain

$$\boldsymbol{F}_{\boldsymbol{x}} = \boldsymbol{\Lambda}(\boldsymbol{q}\ldots\ddot{\boldsymbol{x}} + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{36.3}$$

where $\boldsymbol{\Lambda}$ is the operational space inertial matrix, $\boldsymbol{\Gamma}$ captures centrifugal and Coriolis forces in operational space, and $\boldsymbol{\eta}$ compensates for gravity forces. This equation pertains to a particular operational point. Intuitively speaking, the operational space inertia matrix $\boldsymbol{\Lambda}$ captures the resistance of the operational point to acceleration along and about different axes. More details about operational space control and its relationship to joint space control are provided in Sect. 8.2 (motion control, joint space versus operational space control).

36.2.2 Combined Force and Position Control

Let us consider the example task of controlling the motion of a peg inside a hole. We assume the peg is already inserted into the hole and rigidly connected to the robot's end-effector. We attach the operational frame to a point on the peg so that its z-axis is aligned with the desired direction of motion of the peg inside the hole. To perform this task, the robot has to control the position of the peg along the z-axis of the operational frame and it also has to control contact forces between the peg and the hole to avoid jamming. Thus, the task of moving a peg inside a hole requires that position and force control be seamlessly integrated.

To address contact forces within the operational space framework, we rewrite (36.3) as

$$\boldsymbol{F}_{\boldsymbol{x}} = \boldsymbol{F}_{\mathrm{c}} + \boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{F}_{\mathrm{m}} + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{36.4}$$

where $\boldsymbol{F}_{\mathrm{c}}$ represents the contact forces acting at the end-effector [36.4]. We can replace $\ddot{\boldsymbol{x}}$ with $\boldsymbol{F}_{\mathrm{m}}$ because $\ddot{\boldsymbol{x}}$ is acting on a dynamically decoupled system, i. e., a system that acts like a unit point mass. We can now control forces and motion in the operational frame by selecting the following control structure

$$\boldsymbol{F}_{\boldsymbol{x}} = \boldsymbol{F}_{\mathrm{m}} + \boldsymbol{F}_{\mathrm{c}} \,, \tag{36.5}$$

where

$$\boldsymbol{F}_{\mathrm{m}} = \boldsymbol{\Lambda}(\boldsymbol{q})\,\boldsymbol{\Omega}\boldsymbol{F}'_{\mathrm{m}} + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{36.6}$$

$$\boldsymbol{F}_{\mathrm{c}} = \boldsymbol{\Lambda}(\boldsymbol{q})\,\bar{\boldsymbol{\Omega}}\boldsymbol{F}'_{\mathrm{c}} \,, \tag{36.7}$$

where $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}}$ represent complementary task specification matrices [36.4] that determine along which directions the end-effector is position controlled and along which it is force controlled. By selecting $\boldsymbol{\Omega}$ appropriately, the combination of position and force control can be tailored to the task. In its simplest form, $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}} = I - \boldsymbol{\Omega}$ are diagonal matrices. The i-th diagonal entry of $\boldsymbol{\Omega}$ is 1 if the i-th operational coordinate of the end-effector is position controlled and 0 if it is force controlled. In this simplest case, positions and forces are controlled along axes of the same coordinate frame. The concept of task specification matrix can be extended to include differently oriented coordinate frames for position and force control [36.4].

Once the force $\boldsymbol{F}_{\boldsymbol{x}}$ has been computed as given by (36.5), the corresponding joint torques used to control the robot are computed based on (36.1).

36.2.3 Operational Space Control of Redundant Mechanisms

The full expressiveness of task-level control comes to bear in the context of redundant manipulators. A manipulator is considered redundant with respect to the task it is performing if has more degrees of freedom than required by the task. For example, the task of holding a cup of water only specifies two degrees of freedom,

namely the two rotations about the axes spanning the horizontal plane – this task has two degrees of freedom. The mobile manipulator shown in Fig. 36.2 has ten degrees of freedom, leaving eight redundant degrees of freedom with respect to the task.

The operational space framework for task-level control of redundant manipulators decomposes the overall motion behavior into two components. The first component is given by the task, specified in terms of forces and moments $\boldsymbol{F}_{\text{task}}$ acting at an operational point. This vector $\boldsymbol{F}$ is translated into a joint torque based on (36.2): $\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\boldsymbol{F}_{\text{task}}$. For a redundant manipulator, however, the torque vector $\boldsymbol{\tau}$ is not uniquely specified and we can select from a set of task-consistent torque vectors. The operational space framework performs this selection by considering a secondary task, the so-called posture behavior, making up the second component of the overall motion behavior. Posture behavior can be specified by an arbitrary torque vector $\boldsymbol{\tau}_{\text{posture}}$. To ensure that this additional torque does not affect the task behavior ($\boldsymbol{F}_{\text{task}}$), the torque $\boldsymbol{\tau}_{\text{posture}}$ is projected into the null space $\boldsymbol{N}$ [36.6] of the task Jacobian $\mathbf{J}$. The null space of the Jacobian is the space orthogonal to the one spanned by $\mathbf{J}$ and will be of rank $N_J - k$, where N_J is the number of degrees of freedom of the manipulator and $k = \text{rank}(\mathbf{J})$. The torque $\boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}}$ resulting from the null space projection is task consistent, i. e., it is guaranteed not to affect the behavior of the operational point. Since the vector $\boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}}$ may not lie entirely within the null space of $\mathbf{J}$, however, the execution of the posture behavior cannot be guaranteed.

The operational space task ($\mathbf{J}^{\mathrm{T}}\boldsymbol{F}_{\text{task}}$) and the posture behavior ($\boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}}$) are combined to obtain the general expression for the torque-level decomposition of the overall motion behavior

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\boldsymbol{F}_{\text{task}} \ldots\ldots + \boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}} \,. \tag{36.8}$$

The null-space projection $\boldsymbol{N}$ associated with the task Jacobian $\mathbf{J}$ can be obtained by

$$\boldsymbol{N} = \mathbf{I} - \bar{\mathbf{J}}\mathbf{J} \,, \tag{36.9}$$

where $\mathbf{I}$ is the identity matrix, and $\bar{\mathbf{J}}$ is the dynamically consistent generalized inverse of $\mathbf{J}$, given by

$$\bar{\mathbf{J}} = \mathbf{H}^{-1}\mathbf{J}^{\mathrm{T}}\boldsymbol{\Lambda} \,, \tag{36.10}$$

where $\mathbf{H}$ is the joint space inertia matrix and $\boldsymbol{\Lambda}$ is the operational space inertia matrix for the operational point for which $\mathbf{J}$ is defined. The use of this specific inverse to compute the null-space projection results in the selection of the task-consistent torque vector that minimizes kinetic energy.

36.2.4 Combining Mobility and Manipulation

The operational space framework does not distinguish between degrees of freedom used for mobility and those used for manipulation. The end-effector-centric perspective of this approach considers all degrees of freedom to be in service to position and move the end-effector. An explicit coordination of manipulation and mobility for task-level control is not necessary.

A computationally efficient approach for coordinating manipulation and mobility was introduced in [36.7]. A similar, schema-based coordination of manipulation and mobility in the presence of dynamic obstacles was demonstrated in [36.8]. This approach is based on (36.2) and projects the forces that are derived from various schemas into the configuration space of the robot to determine its motion. Another coordination approach for manipulation and mobility considers a given end-effector path and generates a path of the base that maintains manipulability criteria [36.9]. This underlying representation of mobile manipulators can also be used to generate obstacle avoidance behavior of the base for a given end-effector path [36.10].

The coordination of a nonholonomic mobility platform and a cart-pushing task is addressed in [36.11]. In [36.12], an analysis of the task space for a mobile base with two manipulators is presented. Such a system with two manipulator arms can be viewed as a branching kinematic chain. Section 36.6 discusses extensions to the operational space framework to such branching mechanisms.

36.2.5 Combining Multiple Task Behaviors

The concept of decomposing a redundant robot's overall motion into task and posture behavior can be generalized to an arbitrary number of behaviors [36.13, 14]. Assume a set of n tasks T_i, where T_i has higher priority than T_j if $i < j$. Every task T_i is associated with a force vector $\boldsymbol{F}_i$ and the corresponding joint torque $\boldsymbol{\tau}_i$. These tasks can be performed simultaneously by projecting the joint torque associated with each task into the combined null space of all tasks with higher priority, just as task and posture behavior were combined in the previous section. The null-space projection ensures that the execution of task i does not affect the execution of tasks with higher priority, given by the set $\text{prec}(i)$.

Given a task i, the joint torque consistent with the set of higher priority tasks $\text{prec}(i)$ can be obtained by projecting $\boldsymbol{\tau}_i$ into the combined null space of $\text{prec}(i)$,

$$\boldsymbol{\tau}_{i|\text{prec}(i)} = \boldsymbol{N}^{\mathrm{T}}_{\text{prec}(i)}\boldsymbol{\tau}_i \,, \tag{36.11}$$

where the combined null space $\boldsymbol{N}^{\mathrm{T}}_{\mathrm{prec}(i)}$ is computed as

$$\boldsymbol{N}_{\mathrm{prec}(n)} = \boldsymbol{N}_{n-1}\boldsymbol{N}_{n-2}\cdots\boldsymbol{N}_1 \,. \tag{36.12}$$

We say that $\boldsymbol{\tau}_{i|\mathrm{prec}(i)}$ is the torque for task i given the preceding tasks $\mathrm{prec}(i)$. Applying this torque to the robot will not affect the execution of higher-priority tasks. The original torque vector $\boldsymbol{\tau}_i$ can be computed based on (36.2).

Given the ability to compute a task torque that is consistent with preceding tasks we are able to combine an arbitrary number of tasks into a single motion behavior

$$\boldsymbol{\tau} = \boldsymbol{\tau}_1 + \boldsymbol{\tau}_{2|\mathrm{prec}(2)} + \boldsymbol{\tau}_{3|\mathrm{prec}(3)} + \cdots \boldsymbol{\tau}_{n|\mathrm{prec}(n)} \,, \tag{36.13}$$

where $\boldsymbol{\tau}$ is the torque used to control the robot.

Note that $\boldsymbol{\tau}_1$ is not projected into any null space, since it is the task with the highest priority. Task 1 is therefore performed in the full space defined by the robot's kinematics. If the robot has N degrees of freedom, this space has N dimensions. As the number of tasks performed on the robot increases, the null-space projections $\boldsymbol{N}_{\mathrm{prec}(i)}$ will project the N-dimensional torque vectors associated with lower-priority tasks into a subspace of decreasing dimension. Eventually, the null space will be reduced to the trivial case, when all torque vectors are projected onto the null vector. This prevents lower-priority tasks from being executed. The dimensionality of the null space of tasks $\mathrm{prec}(i)$ can be computed by counting the nonzero singular values [36.6] of $\boldsymbol{N}_{\mathrm{prec}(i)}$, permitting a programmatic assessment of task feasibility.

If the execution of a particular task has to be guaranteed, it should be associated with task 1 in (36.13). Therefore, the highest-priority task is generally used for hard constraints that under no circumstances should be violated. These constraints may include contact constraints, joint limits, and balancing in the case of humanoid robots, for example (Chap. 67).

The projection of (36.11) maps the torque $\boldsymbol{\tau}_i$ of task i into the null space of all preceding tasks. This projection also scales the vector, which may significantly impair the execution of task i. This effect can be reduced by scaling the desired accelerations $\ddot{\boldsymbol{x}}$ at the operational point with the task-consistent inertia matrix $\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}$, given by

$$\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)} = \left(\mathbf{J}_{i|\mathrm{prec}(i)}\mathbf{H}^{-1}\mathbf{J}^{\mathrm{T}}_{i|\mathrm{prec}(i)}\right)^{-1} \,, \tag{36.14}$$

where $\mathbf{H}$ is the joint space inertia matrix and

$$\mathbf{J}_{i|\mathrm{prec}(i)} = \mathbf{J}_i\boldsymbol{N}_{\mathrm{prec}(i)} \tag{36.15}$$

is the task Jacobian consistent with all preceding tasks. This Jacobian projects operational space forces into joint torques that do not cause any acceleration at the operational points of tasks in $\mathrm{prec}(i)$. The task-consistent inertia matrix $\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}$ then captures the inertia perceived at the operational point when its motion is limited to the subspace consistent with all preceding tasks.

Using the task-consistent inertia matrix, we can obtain the task-consistent operational space dynamics by projecting the joint space dynamics into the space defined by the dynamically consistent inverse of $\mathbf{J}_{i|\mathrm{prec}(i)}$. Inverting (36.2) and replacing $\boldsymbol{\tau}$ based on (36.2) we obtain

$$\bar{\mathbf{J}}^{\mathrm{T}}_{i|\mathrm{prec}(i)}\left(\mathbf{H}(\boldsymbol{q}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{\tau}_g(\boldsymbol{q}) = \boldsymbol{\tau}_{k|\mathrm{prec}(i)}\right) , \tag{36.16}$$

which yields the task-consistent equivalent to (36.3)

$$\boldsymbol{F}_{i|\mathrm{prec}(i)} = \boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}\ddot{\boldsymbol{x}} + \boldsymbol{\Gamma}_{i|\mathrm{prec}(i)}\dot{\boldsymbol{x}} + \boldsymbol{\eta}_{i|\mathrm{prec}(i)} \,, \tag{36.17}$$

where $\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}$ is the task-consistent operational space inertial matrix, $\boldsymbol{\Gamma}_{i|\mathrm{prec}(i)}$ captures centrifugal and Coriolis forces in operational space, and $\boldsymbol{\eta}_{i|\mathrm{prec}(i)}$ compensates for gravity forces. Equation 36.17 can be used to control a task at a particular operational point in a task-consistent manner, i. e., without causing acceleration at the operational points associated with the tasks in $\mathrm{prec}(i)$. Note that in practice the terms $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ and $\boldsymbol{\tau}_g(\boldsymbol{q})$ can be computed in the joint space; they only need to be accounted for once in the case of multiple tasks.

Referring back to Fig. 36.3, we can analyze the effectiveness of the operational space framework. The control of operational points directly addresses force and position constraints on motion in manipulation tasks. Additional behaviors with lower priority than the task can be used to ensure the maintenance of kinematic and dynamic constraints, implement posture behavior, or perform reactive obstacle avoidance based on artificial potential fields [36.15]. The control of all of these aspects of manipulation tasks is computationally efficient. As a consequence, the operational space framework is able to maintain these motion constraints while satisfying feedback frequency requirements. However, the operational space framework does not include the consideration of global motion constraints, such as those imposed by the connectivity of the work space. Therefore, the operational space framework is susceptible to local minima, similarly to the artificial potential field method. Such local

minima can occur for all of the aforementioned motion constraints and would prevent the robot from accomplishing its task. To overcome this problem, planning methods have to be applied (Chap. 7 and Sect. 36.3).

The control basis approach [36.16] provides an alternative, compact formalism for combining multiple behaviors. Each individual behavior is described by a controller, represented by a potential function $\phi \in \Phi$, sensors $\sigma \in \Sigma$, and effectors $\tau \in \Upsilon$. A particular controller ϕ can be parameterized with sensors and effectors. A particular instance of a behavior can be written as $\phi|_{\tau}^{\sigma}$. These controllers can be combined using null-space projections, similar to (36.13). In the control basis framework, (36.13) would be written more compactly as

$$\phi_n \triangleleft \cdots \triangleleft \phi_3 \triangleleft \phi_2 \triangleleft \phi_1 \,, \tag{36.18}$$

where we abbreviate $\phi_i|_{\tau_i}^{\sigma_i}$ with ϕ_i. The relation $\phi_i \triangleleft \phi_j$ means that ϕ_i is projected into the null space of ϕ_j. We say that ϕ_i is executed *subject to* ϕ_j. If more than two behaviors are combined, as shown in (36.18), each controller ϕ_i is projected into the combined null space of all superior tasks.

The control basis represents a general framework for combining controllers and is not limited to the notion of task-level control. In addition to the aforementioned formalism for representing combinations of controllers, the control basis also encompasses discrete structures that define transitions between combinations of controllers. Such transitions can be specified by the user or learned with techniques from reinforcement learning [36.16]. The control basis can compose multiple task-level controllers to implement manipulation tasks that cannot be described in terms of a single task-level controller.

36.3 Manipulation Planning

In order to appreciate the computational challenges inherent to gross motion planning for manipulation, it is useful to adopt a mathematical formalism for analysis. The chapter on planning (Chap. 7) introduced the *configuration space* (*C-space*), which represents the space of allowable transformations of the robot. In general, the set of all allowable transformations forms an n-dimensional *manifold*, in which n is the total number of degrees of freedom (DOF) of the mechanical system. For a robotic manipulator, the configuration space is defined by its kinematic structure (Chap. 2). One common class of manipulators is *serial-chain* manipulators, which consist of a series of revolute or prismatic joints connected to form a linear kinematic chain. The discussion of manipulation planning in this section focuses on configuration spaces arising from single serial-chain manipulators. However, the mathematical formulation generalizes to more complex robots such as humanoids (Fig. 36.4; Chap. 67), whose upper body consists of two cooperating manipulator arms attached to a movable base (Chap. 39). The model considered here greatly simplifies the problems of grasping, stability, friction, mechanics, and uncertainties and instead focuses on the geometric aspects. In this section, we will consider a simple example of a manipulation task that involves a classic pick-and-place operation. Section 36.4 further considers grasping and contact states in the context of assembly operations, and Sect. 36.5 gives an overview of techniques to unify feedback control and planning for more complex manipulation tasks.

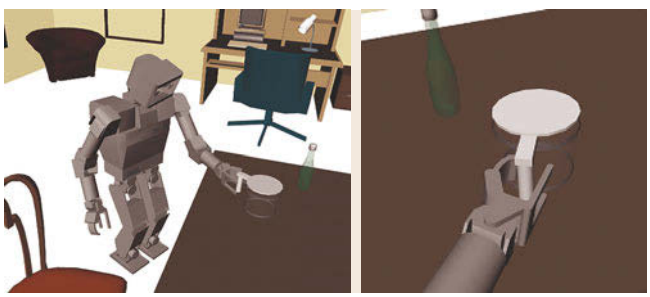


Fig. 36.4 Simulation of the H6 humanoid robot manipulating an object

36.3.1 Configuration Space Formalism

The basic motion planning problem is defined as finding a path in the configuration space connecting a start and a goal configuration while avoiding static obstacles in the environment (Chap. 7). Manipulation planning involves additional complexity because manipulation tasks fundamentally require contact with objects in the environment. When objects are grasped, regrasped, and moved, the structure and topology of the free configuration space can change drastically. However, manipulation planning shares the same basic theoretical foundations and computational complexity as classical path planning, such as PSPACE-hardness (Chap. 7).

To be more precise, we will utilize a formalization for manipulation planning that is primarily due to *Alami*

et al. [36.17], and adapted in *LaValle*'s book [36.18]. Our goal is to gain understanding and intuition regarding the search space for manipulation tasks and the different situations that can arise. Throughout this discussion, bear in mind that the geometry and relative positions of the manipulator, the manipulated objects, and the obstacles in the workspace determine both the geometry and topological structure of the free configuration space. Within this framework, manipulation planning can be intuitively placed in the context of planning for *hybrid systems* [36.19]. Hybrid systems involve a mixture of continuous variables and discrete *modes*, whose transitions are defined by switching functions (for a discussion of hybrid systems in a planning context, see Chap. 7 of *LaValle*'s book [36.18]). In the case of manipulation planning, the effect of grasping, repositioning, and releasing objects in the environment corresponds to switching between different continuous configuration spaces.

Let us consider a simple example of a simple *pick-and-place manipulation task* that requires a robot manipulator $\mathcal{A}$ to reposition a single movable rigid object (part) $\mathcal{P}$ from its current location in the workspace to some desired goal location. Note that accomplishing the task only requires that the part $\mathcal{P}$ reach the goal location, without regard to precisely *how* the manipulator does so. Because the part cannot move by itself, it must either be transported by the robot or be placed at rest in some stable intermediate configuration. There are some configurations in which the robot is able to grasp and move the part to other locations in the environment. There are also forbidden configurations in which the robot or the part collides with obstacles in the environment. The solution of a manipulation planning problem consists of a sequence of subpaths for the robot moving alone or moving while grasping the part. The primary constraints on the allowable motions are:

1. The robot manipulator must not collide with any obstacles in the environment while reaching for, grasping, or regrasping the part.
2. The robot must adequately grasp the part while transporting it (for details on grasping and metrics for stable grasps see Chap. 38).
3. The robot manipulator and movable part must not collide with any obstacles in the environment while the part is being transported.
4. If the part is not being transported by the robot, it must be placed at rest in some stable intermediate placement.
5. The part must end up at the desired goal location in the workspace.

The two modes of operation give rise to the hybrid systems formulation: either the robot moves alone, or the robot moves while grasping the part. In the literature, motions of the robot holding the object at a fixed grasp are called *transfer paths*, and motions of the robot while the part stays at a stable placement are called *transit paths* [36.17, 20, 21]. Solving a manipulation planning problem involves searching for a connected sequence of *transit* and *transfer* paths for the manipulator that will accomplish the task. For an autonomous robot, the ideal is to utilize planning algorithms that will automatically produce correct sequences of transfer and transit paths separated by grasping and ungrasping operations.

Admissible Configurations

Building on the notation introduced in Chap. 7, we define a robot manipulator $\mathcal{A}$ with n degrees of freedom operating in a Euclidean *world* or *workspace*, $\mathcal{W} = \mathbb{R}^N$, in which $N = 2$ or $N = 3$. Let C^A represent the n-dimensional C-space of $\mathcal{A}$, and q^A be a configuration. C^A is called the *manipulator configuration space*. Let $\mathcal{P}$ denote a movable part, which is a solid rigid body modeled with geometric primitives. We will assume that $\mathcal{P}$ is allowed to undergo rigid-body transformations. This gives rise to a *part configuration space*, $C^P = SE(2)$ or $C^P = SE(3)$. Let $q^P \in C^P$ denote a part configuration. The area or volume in the workspace occupied by the transformed part model is denoted by $\mathcal{P}(q^P)$.

We define the combined (robot and part) configuration space C as the Cartesian product

$$C = C^A \times C^P \,,$$

in which each configuration $q \in C$ is of the form $q = (q^A, q^P)$. Note that the set of admissible configurations for manipulation planning purposes is more restrictive than simply C_{free}, the space free of collisions with obstacle regions as defined in Chap. 7. We must incorporate the constraints on the allowable motions by removing inadmissable configurations. Figure 36.5 illustrates examples of some of the important subsets of C for manipulation planning.

We start by removing all configurations that involve collisions with obstacles. Let the subset of configurations in which the manipulator collides with obstacles be given by

$$C^A_{\text{obs}} = \{(q^A, q^P) \in C | \mathcal{A}(q^A) \cap \mathcal{O} \neq \emptyset\} \,.$$

We also want to remove configurations for which the part collides with obstacles. However, we will allow the surface of the part to make contact with obstacle surfaces. This happens for example, when the part is

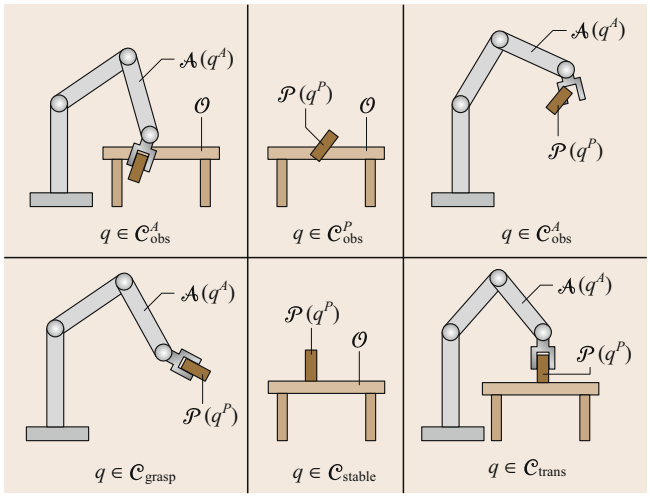


Fig. 36.5 Examples of the various configuration space subsets defined for manipulation planning

resting on a shelf or table, or when a peg-shaped part is being inserted into a hole. Therefore, let

$$C^P_{\text{obs}} = \{(q^A, q^P) \in C | \text{int}(\mathcal{P}(q^P)) \cap \mathcal{O} \neq \emptyset\}$$

denote the open set for which $\mathcal{O}$ intersects the interior volume $\text{int}(\mathcal{P}(q^P))$ of the part for any configuration q^P. If the interior of the part penetrates $\mathcal{O}$, then clearly these configurations should be avoided.

We can now define $C \setminus (C^A_{\text{obs}} \cup C^P_{\text{obs}})$, which is the set of all configurations in which both the robot and the part do not inappropriately collide with $\mathcal{O}$. Next consider the interaction between $\mathcal{A}$ and $\mathcal{P}$. The manipulator must be allowed to make contact with the surface of the part, but penetration is once again not allowed. Therefore, let all configurations in which the interior of the part overlaps with the robot geometry be defined as

$$C^{PA}_{\text{obs}} = \left\{(q^A, q^P) \in C | A(q^A) \cap \text{int}(\mathcal{P}(q^P)) \neq \emptyset\right\} .$$

Finally, we are able to define

$$C_{\text{adm}} = C \setminus \left(C^A_{\text{obs}} \cup C^P_{\text{obs}} \cup C^{PA}_{\text{obs}}\right) ,$$

which represents the set of configurations remaining after subtracting away all unwanted configurations. We call this the set of *admissible configurations*.

Stable and Grasped Configurations

By definition, all configurations in the set C_{adm} are free of penetrative collisions between the robot, the part, and obstacles in the environment. However, many of these configurations define the part as floating in space or on the verge of falling. We now consider two important subsets of C_{adm} that are used in manipulation planning (Fig. 36.5). Let $C^P_{\text{stable}} \subseteq C^P$ denote the subset of *stable part configurations*, which are configurations at which the part can safely rest without any forces being applied by the manipulator. Examples of stable configurations include the part resting on a table or inserted into a larger assembly of other parts. The criteria for inclusion into the set of stable configurations depends on properties such as the part geometry, friction, mass distribution, and contacts with the environment. Our formalism does not consider these issues directly, but assumes that some method for evaluating the stability of a part configuration is available. Given C^P_{stable}, we define $C_{\text{stable}} \subseteq C_{\text{adm}}$ to be the corresponding stable configurations of the part–robot system

$$C_{\text{stable}} = \left\{(q^A, q^P) \in C_{\text{adm}} | q^P \in C^P_{\text{stable}}\right\} .$$

The other important subset of C_{adm} is the set of all configurations in which the robot is grasping the part and is capable of manipulating it according to some defined criteria. Let $C_{\text{grasp}} \subseteq C_{\text{adm}}$ denote the set of *grasp configurations*. For every configuration, $(q^A, q^P) \in C_{\text{grasp}}$, the manipulator touches the part, which implies that $\mathcal{A}(q^A) \cap \mathcal{P}(q^P) \neq \emptyset$. Just as before, penetration between the robot and part geometry is not allowed because $C_{\text{grasp}} \subseteq C_{\text{adm}}$. In general, many configurations at which $\mathcal{A}(q^A)$ contacts $\mathcal{P}(q^P)$ will not necessarily be in C_{grasp}. The criteria for a configuration to lie in C_{grasp} depends on the particular characteristics of the manipulator, the part, and the contact surfaces between them. For example, a typical manipulator would not be able to pick up a part by making contact at only a single point. (For additional information, see Sect. 36.4 for contact state identification, Chap. 37, and grasping criteria, such as *force closure* models in Chaps. 38 and 19).

For any robot and part configuration $q = (q^A, q^P) \in C$ that we consider for manipulation planning we must always ensure that either $q \in C_{\text{stable}}$ or $q \in C_{\text{grasp}}$. We therefore define $C_{\text{free}} = C_{\text{stable}} \cup C_{\text{grasp}}$, to reflect the subset of C_{adm} that is permissible for manipulation planning. In the context of hybrid systems, this gives rise to the two distinct *modes* for manipulation planning: the *transit mode* and the *transfer mode*. Recall that in the transit mode, the manipulator is not carrying the part (i. e., only the robot moves), which requires that $q \in C_{\text{stable}}$. In the transfer mode, the manipulator carries the part (i. e., both the robot and the part move), which requires that $q \in C_{\text{grasp}}$. Based on these conditions, the only way the mode can change is if $q \in C_{\text{stable}} \cap C_{\text{grasp}}$. When the manipulator is in these configurations, it has two available actions: (1) continue to grasp and move the part, or (2) release the part in its currently stable configuration. In all other configura-

tions the mode remains unchanged. For convenience, we define $C_{\text{trans}} = C_{\text{stable}} \cap C_{\text{grasp}}$ to denote the set of *transition configurations*, which are the places in which the mode may change and the robot may grasp or release the part.

Manipulation Planning Task Definition

Finally, the basic pick-and-place manipulation planning task can now be defined. An *initial part configuration*, $q^P_{\text{init}} \in C^P_{\text{stable}}$, and a *goal part configuration*, $q^P_{\text{goal}} \in C^P_{\text{stable}}$, are specified. Recall that this high-level task specification is only defined in terms of repositioning the part, without regard to how the manipulator actually accomplishes the task. Let $C_{\text{init}} \subseteq C_{\text{free}}$ be the set of all configurations in which the part configuration is q^P_{init}

$$C_{\text{init}} = \left\{(q^A, q^P) \in C_{\text{free}} | q^P = q^P_{\text{init}}\right\} .$$

We define $C_{\text{goal}} \subseteq C_{\text{free}}$ similarly as the set of all configurations in which the part configuration is q^P_{goal}

$$C_{\text{goal}} = \left\{(q^A, q^P) \in C_{\text{free}} | q^P = q^P_{\text{goal}}\right\} .$$

If the initial and final configurations of the manipulator are specified, then let $q_{\text{init}} = (q^A_{\text{init}}, q^P_{\text{init}}) \in C_{\text{init}}$ and $q_{\text{goal}} = (q^A_{\text{goal}}, q^P_{\text{goal}}) \in C_{\text{goal}}$. The objective of the planner is to compute a path τ such that

$$\tau : [0, 1] \mapsto C_{\text{free}} ,$$
$$\tau(0) = q_{\text{init}}; \tau(1) = q_{\text{goal}} .$$

If the initial and final positions of the manipulator are unspecified, we implicitly define the objective of the planner as computing a path $\tau : [0, 1] \mapsto C_{\text{free}}$ such that $\tau(0) \in C_{\text{init}}$ and $\tau(1) \in C_{\text{goal}}$. In either case, a solution is an alternating sequence of transit paths and transfer paths, whose names follow from the mode. In between each transfer path, the part is placed in a stable intermediate configuration while the manipulator moves alone (transit path) in order to regrasp the part. This sequence continues until the part is ultimately placed to rest in its final goal configuration. This is depicted in Fig. 36.6.

36.3.2 Example of a Three-DOF Planar Manipulator

For manipulators with three or fewer degrees of freedom (DOF), we can construct visualizations of the C-space in order to gain an intuition of the structure. Throughout this section, we will use the example of a three-DOF serial-chain manipulator whose end-effector operates a planar workspace ($\mathbb{R}^2$).

Figure 36.7 shows a simple redundant robot arm originally illustrated in [36.22]. The robot has three revolute joints with parallel axes and joint limits of $-\pi, \pi$ radians, providing three degrees of freedom. Since the end-effector of the robot can only reach positions in a plane, the arm is redundant in this plane and since it has only three DOF and limits imposed on the joint angles, C can be visualized as a cube with an edge length of 2π. Generally, the C-space manifold of a manipulator with n revolute joints is homeomorphic to an n-dimensional *hypertorus* if no joint limits are present, and homeomorphic to an n-dimensional *hypercube* if joint limits are present.

Recall that obstacles in the workspace are mapped to regions in the C-space called C-obstacles (Chap. 7), which represent the set of all joint configurations of the robot that cause an intersection of the geometry of the robot and the obstacle

$$C^A_{\text{obs}} = \left\{(q^A, q^P) \in C | A(q^A) \cap \mathcal{O} \neq \emptyset\right\} .$$

In general, the structure of C^A_{obs} can be highly complex and involve multiple connected components of C. An example visualization of a configuration space for the planar three-DOF manipulator corresponding

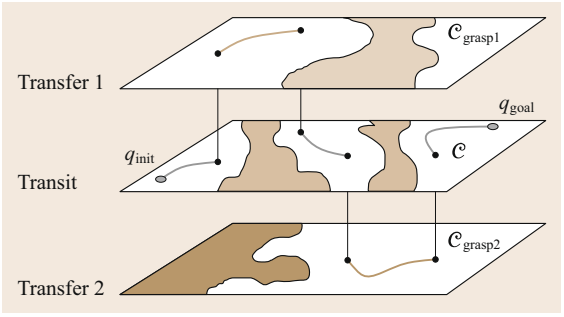


Fig. 36.6 Manipulation planning involves searching for a sequence of transfer and transit paths in a set of hybrid continuous configuration spaces. In this example, transfer paths in different C-spaces arise from different ways of rigidly grasping a part

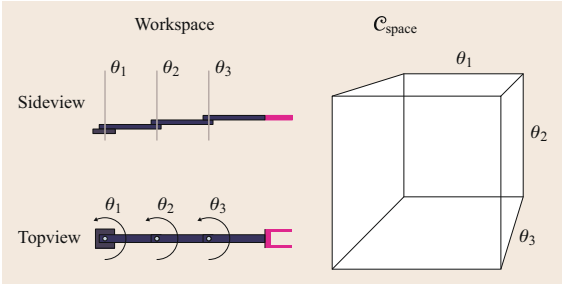


Fig. 36.7 Planar 3 DOF manipulator

to a workspace with two differently colored box-shaped obstacles is given in Fig. 36.8. The C-obstacles are illustrated using the same color as the corresponding box-shaped obstacle in the workspace. The red, green, and blue coordinate axes correspond to the three joint angles $\theta_{\{1,2,3\}}$ of the manipulator, respectively.

Self-collisions induce special C-obstacle regions that do not involve environment geometry, but only the geometric interference of the links of the robot itself. The robot shown in Fig. 36.7 is, by design, self-collision-free. Figure 36.9 shows a modified version of the manipulator, where self-collision is actually possible, and a visualization of the corresponding C-obstacles.

C-obstacles can cover large parts of the configuration space and even cause a complete disconnection between different components of C_{free}. Figure 36.10 shows how the addition of only a small post as an obstacle can cause a wide range of configurations to produce collisions and disconnect regions of C_{free}: the C-obstacle corresponding to the post forms a *wall* of configurations parallel to the θ_2–θ_3-plane. This wall represents the range of values for θ_1 for which the first link of the manipulator intersects with the post, splitting C_{free} into two disconnected components. Having disconnected components $C_1, C_2 \in C_{\text{free}}$ means that there is no collision-free path that starts in C_1 and ends in C_2.

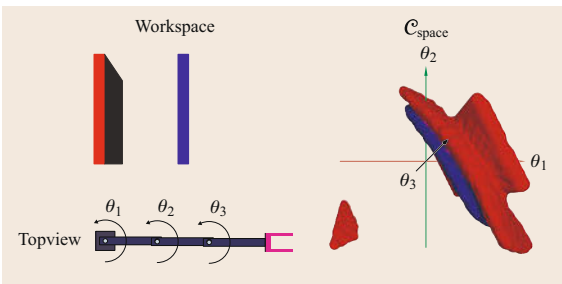


Fig. 36.8 Planar 3 DOF manipulator in a workspace with box-shaped obstacles and associated configuration space obstacles

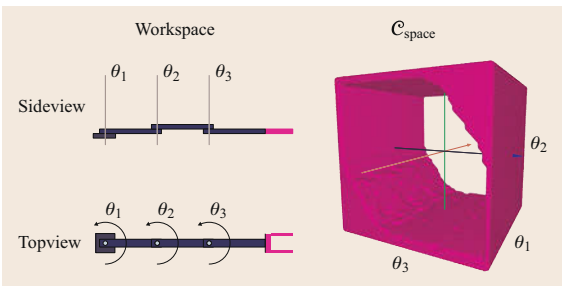


Fig. 36.9 Non-planar 3 DOF manipulator and corresponding self-collision C-obstacle

36.3.3 Inverse Kinematics Considerations

The typical formulation of a path planning problem requires a goal configuration q_{goal} as input (Chap. 7). For the case of manipulation planning, this configuration is usually computed from the desired workspace pose of the end-effector by an inverse kinematics (IK) solver (Sect. 2.7). Apart from special cases, there currently exist no known analytical methods for solving the inverse kinematics of a general redundant mechanism (greater than six degrees of freedom). Iterative, numerical techniques (such as Jacobian-based methods or gradient-based optimization) are typically used to calculate solutions in this case. Note that the numerical computation of inverse kinematics solutions can suffer from poor performance or even nonconvergence, depending on the quality of the initial guess, the distribution of singularities in the C-space or a combination of these effects (Sect. 2.7).

Most IK solvers are only able to compute one configuration for a given end-effector pose. For a redundant manipulator, however, an infinite, continuous range of configurations that cause the end-effector to assume the desired pose usually exists. In the context of path planning, this means that the IK solver will select one out of an infinite number of configurations as q_{goal}. Note that in some cases, the selected q_{goal} might not be collision-free, i. e., it is not part of C_{free}. Such a configuration cannot be used as a goal for computing a global reaching strategy because no solution exists for the corresponding path planning query in the classical sense (Chap. 7).

Another possibility is that the selected q_{goal} might be disconnected from q_{init}. In this case, q_{goal} is unsuitable for planning despite the fact that it is a member of C_{free}, because it is in a component of C_{free} that is disconnected from the component in which q_{init} lies. Consider the planar manipulator shown in Fig. 36.7.

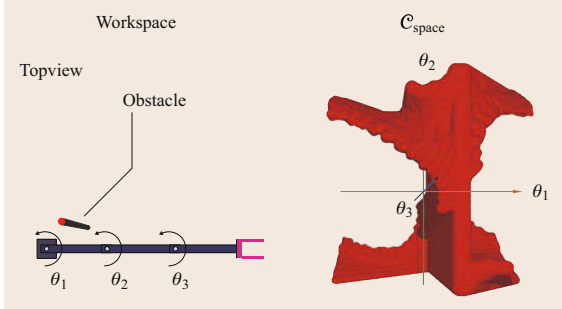


Fig. 36.10 Workspace mapping to a configuration space with topologically disconnected components

Figure 36.11 shows two possible solutions for a certain end-effector pose. The corresponding configurations $q_a = (0, \pi/2, 0)$ and $q_b = (\pi/2, -\pi/2, \pi/2)$ are marked by green spheres in the C-space visualization. In this case, the IK solver might select one of the two solutions as q_{goal}. If the same obstacle as in Fig. 36.10 is added to the workspace, the two solutions will lie in disconnected components of C_{free}, as shown in Fig. 36.12.

Let $q_{init} = (0, 0, 0)$ be the zero configuration, marked by the point of intersection of the C-space axes in Fig. 36.12. Let the component of C_{free} in which it lies be denoted by C_1 and the other component by C_2. Since $q_b \in C_2$ and $C_1 \cap C_2 = \emptyset$, no solution for the planning problem can exist if the IK solver selects q_b as q_{goal}. This means that every planning attempt with this q_{goal} will invariably fail.

Since computing a complete representation of the connectivity of C is a problem of similar complexity to path planning itself, it is virtually impossible to detect a disconnection between q_{goal} and q_{init} before the planning attempt has failed.

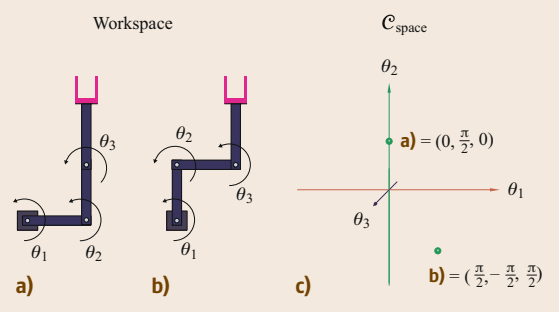


Fig.36.11a–c Two inverse kinematics solutions for the same end-effector pose of a planar 3 DOF arm; **(a)** solution 1, **(b)** solution 2, **(c)** solution shown in C space

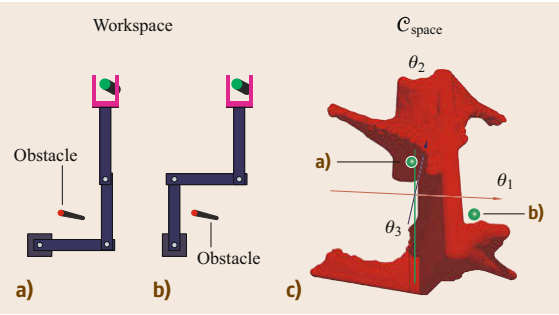


Fig.36.12a–c Two inverse kinematics solutions lying in disconnected components of C_{free} due to a small obstacle in the workspace; **(a)** solution 1, **(b)** solution 2, **(c)** the two solutions shown in C space, which are in disconnected components

Recall that the workspace goal for a classical manipulation planning problem is defined by a desired end-effector pose computed using inverse kinematics. Note however, that the purpose of a manipulation task may not require moving the end-effector to that particular location, but rather moving it to *any* feasible location which enables the desired manipulation task to be solved. Thus, it becomes important to devise computational methods and control schemes that are focused around the task, and allow some flexibility and freedom with regards to the end-effector pose and manipulator configuration. This is one of the motivations and advantages of task space or operational space control techniques (Sect. 8.2).

As an example, consider the task of grasping a cylindrical part with the planar three-DOF manipulator. Figure 36.11 shows two possible solution configurations yielding the same end-effector pose. The addition of obstacles to this workspace may cause one of the configurations to become disconnected from the zero configuration. These two configurations, however, may not be the only valid configurations that allow the manipulator to grasp the cylindrical part. Figure 36.13a shows several other configurations that allow the manipulator to grasp the part. By interpolating between these, an infinite number of different workspace poses that provide solutions to the grasping problem is generated. The inverse kinematics solutions associated with these workspace poses are contained in the continuous subset $C_{goal} \subseteq C$, as shown in Fig. 36.13b, which represents the actual solution space to the planning problem for this reaching task.

Adding the obstacle from Fig. 36.10 to the workspace of this manipulation task produces the configuration space depicted in Fig. 36.14. Note that the C-obstacle corresponding to collisions between the manipulator and the cylinder itself has been disregarded in this visualization for clarity. Approximately half of the

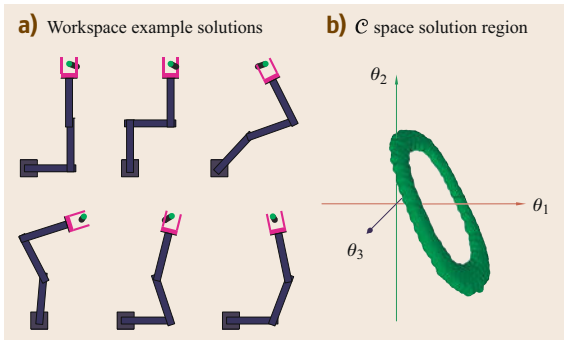


Fig. 36.13 **(a)** Multiple arm configurations that allow the manipulator to grasp the cylinder and **(b)** C-space visualization of the set of solution configurations

possible solutions are disconnected from the zero configuration and a quarter are in collision, which leaves only one quarter as reachable and therefore suitable candidates for q_{goal} in a classical planning query.

36.3.4 Continuum Manipulation

As introduced in Chap. 20, a manipulator with far more degrees of freedom than that of an object to be manipulated is a hyperredundant manipulator. Hyperredundant manipulators can be further classified into vertebrate-like rigid-link manipulators, such as snakes, and invertebrate-like continuum manipulators, such as octopus arms or elephant trunks. Many vertebrate-like manipulators inherit the kinematics of articulated arms, and like the articulated arms, they are often used as carriers of the actual manipulators: hands or grippers, such as in surgical applications.

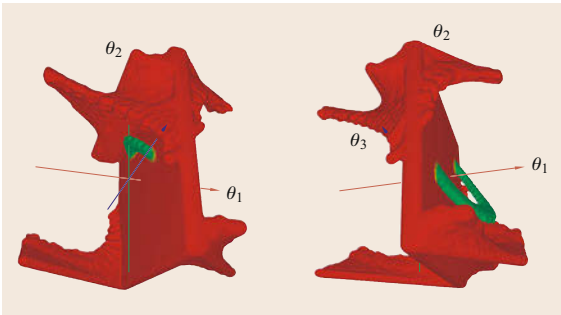


Fig. 36.14 Two views of the set of goal configurations (*green*) for grasping a cylindrical part in a disconnected *C*-space

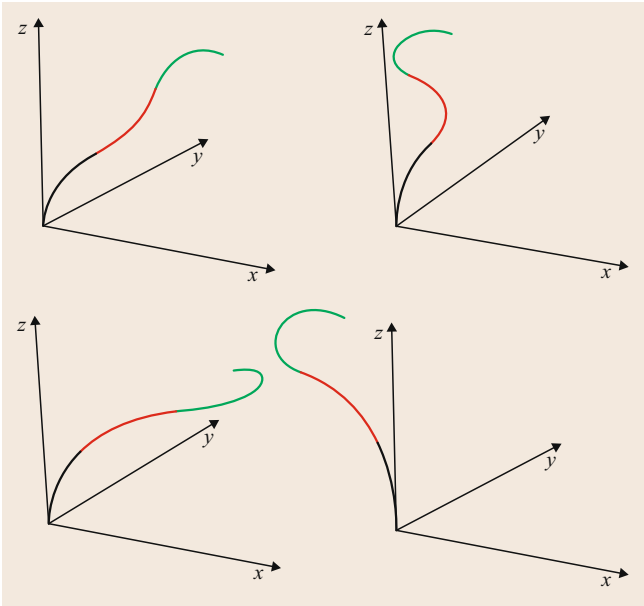


Fig. 36.15 Different configurations of the OctArm manipulator

Continuum manipulators [36.23], on the other hand, consist of deformable sections and are often used for whole-arm manipulation without a divide between the *arm* and *hand/gripper*, taking advantage of their compliance and flexibility. They can be viewed as the extreme case of hyperredundancy but are fundamentally different from articulated arms. A *grasping* configuration for a continuum manipulator affects the whole arm as it wraps around an object. The smooth and compliant nature of a continuum manipulator allows it to gently interact with the object by adapting its shape to the object it wraps. The OctArm [36.24], with multiple constant-curvature sections, is representative of the general class of continuum manipulators.

Although each section i of a continuum manipulator is deformable, and thus has infinite degrees of freedom theoretically, it often has three controllable degrees of freedom, which describes the configuration of the section as a truncated torus when undeformed or a cylinder if the curvature is zero. length s_i, curvature κ_i, and the orientation angle ϕ_i between section $i-1$ and section i. By changing those three variables for each section, the configuration of the whole manipulator can be changed. Figure 36.15 shows different configurations of a three-section OctArm.

Planning for continuum, whole-arm manipulation is significantly different from manipulation and grasping planning for conventional articulated arms with hands/grippers. Here the inverse kinematics problem is to find end positions of each section for the manipulator, given the configuration variables of the section [36.25]. However, planning motion for whole-arm manipulation has to consider directly a mixture of configuration variables and Cartesian-space variables (i. e., the end position of each section), because the shapes and grasping capabilities of the end-effector (which can be viewed as the portion of the continuum manipulator contacting the target object) depend on both.

Given a target object, its location and size decide if there exists a configuration that a continuum manipulator can wrap around the object to grasp it. If wrapping around the object along one circle is sufficient, that bounding circle of the object centered at position c with radius r_c determines the k ($k \leq n$) sections that are needed to wrap it, starting from the tip n-th section backward. The poses of the remaining $n-k$ sections must satisfy the intersection constraints to form feasible whole-arm configurations for grasping [36.26]. A feasible grasping configuration can also be determined heuristically as an integral part of collision-free motion planning [36.27]. Multiple grasping solutions provide flexibility to avoid obstacles (VIDEO 357).

A more practical approach [36.28] is to first make a contact between the manipulator and the target object at the m-th section ($m < n$), and then generate wrapping configurations for the m to n sections of the manipulator one by one without overlap to form a tight, spiral wrap (i. e., along multiple object bounding circles). In a cluttered environment, progressive wrapping of an object can be achieved through moving all arm sections together led by the tip of the arm to surroun. the object [36.29], which requires minimum free space for the arm to manuever. Those approaches can generate a better force-closure grasp and facilitate sensing-based manipulation based on contact detection.

Finally, strategies for planning continuum manipulation could also benefit whole-arm manipulation with vertebrate-like hyperredundant manipulators in that the planned spatial curves for poses of a continuum manipulator could be used as desired shapes for a hyperredundant manipulator, such as a snake robot, to fit its joint configurations. Indeed, curve design and fitting for hyperredundant manipulators has long been introduce. for planning manipulator poses [36.30].

In this section, we have covered the basics of manipulation planning, developed a mathematical formulation for the kinds of search spaces that arise, and discussed some of the important geometric and topological considerations, as well as different types of manipulators and manipulation. Section 36.6 provides an overview of additional topics and extensions with pointers to further reading.

36.4 Assembly Motion

Assembly or an *assembly task* defines the process of putting together manufactured parts to make a complete product, such as a machine. It is a major operation in the manufacturing process of any product. Assembly automation with robots aims to reduce cost and increase the quality and efficiency of the operation. In environments hazardous to humans, such as in space, having robots perform assembly tasks could save human lives. Assembly has long been not only an important but also one of the most challenging applications for robotics. There are many significant research issues related to the broad scope of assembly automation, from design for assembly [36.31] to tolerance analysis [36.32], assembly sequence planning [36.33], fixture design [36.34], etc. This section is only focused on the issue of robotic motion for assembly.

The concerned *assembly motion* is that of a robot manipulator holding a part and moving it to reach a certain *assembled state*, i. e., a required spatial arrangement or contact against another part. The main difficulty of assembly motion is due to the requirement for high precision or low tolerance between the parts in an assembled state. As a result, the assembly motion has to overcome uncertainty to be successful. *Compliant motion* is defined as motion constrained by the contact between the held part and another part in the environment. As it reduces uncertainty through reducing the degrees of freedom of the held part, compliant motion is desirable in assembly.

Consider the peg-in-hole insertion example introduced earlier (Fig. 36.2). If the clearance between the peg and the hole is very small, the effect of uncertainty will most likely cause a downward insertion motion of the peg to fail, i. e., the peg ends up colliding with the entrance of the hole in some way without reaching the desired assembled state. Therefore, a successful assembly motion has to move the peg out of such an unintended contact situation and lead it to reach the desired assembled state eventually. To make this transition, compliant motion is preferred. Often a sequence of contact transitions via compliant motion is necessary before the desired assembled state can be reached. Figure 36.16 shows a typical sequence of contact transitions for the peg-in-hole task.

Assembly motion strategies that incorporate compliant motion can be broadly classified into two groups: *passive compliance* and *active compliant motion*, and both groups of strategies require certain informa-

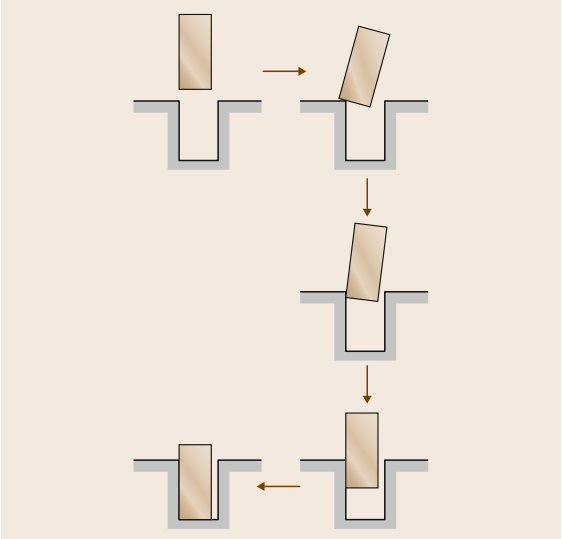


Fig. 36.16 A sequence of contact transitions for the peg-in-hole insertion task

tion characterizing *topological contact states* between parts.

36.4.1 Topological Contact States

When a part A contacts a part B, the configuration of A (or B) is a *contact configuration*. Often a set of contact configurations share the same high-level contact characteristics. For example, *a cup is on the table* is a high-level description shared by all the contact configurations of the cup when its bottom are on the table. Such a description is often what really matters in assembly motion as it characterizes a spatial arrangement that could be either an assembled state or just a *contact state* between a part and another part.

For contacting polyhedral objects, it is common to describe a *contact state* topologically as a set of *primitive contacts*, each of which is defined by a pair of contacting surface elements in terms of faces, edges, and vertices. Different contact state representations essentially differ only in how primitive contacts are defined. One common representation [36.35] defines primitive contacts as point contacts in terms of vertex–edge contacts for two-dimensional (2-D) polygons, and vertex–face and edge–edge contacts for three-dimensional (3-D) polyhedra. Another representation [36.36] defines a primitive contact as between a pair of *topological surface elements* (i. e., faces, edges, and vertices). Here a contact primitive can characterize a contact region that is either a point, a line segment, or a planar face, unlike the point-contact notion. From the viewpoint of contact identification via sensing, however, both representations can result in states that are different by definition but indistinguishable in identification due to uncertainties. Figure 36.17 shows such an example.

The notion of a *principal contact* [36.37, 38] presents a high-level primitive contact that is more robust to recognition. A *principal contact* (PC) denotes a contact between a pair of topological surface elements that are not boundary elements of other contacting topological surface elements. The boundary elements of a face are the edges and vertices bounding it, and the boundary elements of an edge are the vertices bounding it. Different PCs between two objects correspond to different degrees of freedom of the objects, which often also correspond to significant differences in the contact forces and moments. As shown in Fig. 36.17, the indistinguishable states in terms of the other contact primitives are grouped as a single contact state in terms of PCs. Thus, there are also fewer contact states in terms of PCs, leading to a more concise characterization of contact states. In fact, every contact state between two convex polyhedral objects is described by a single PC.

36.4.2 Passive Compliance

Passive compliance refers to strategies that incorporate compliant motion for error correction during the assembly motion without requiring active and explicit recognition and reasoning of contact states between parts.

Remote Center Compliance

In the 1970s, a *remote center compliance* (RCC) device was developed to assist high-precision peg-in-hole insertion [36.39–41]. The RCC is a mechanical spring structure used as a tool attached to the end-effector of a robot manipulator to hold a round peg when it is inserted into a round hole. The RCC is designed to have high stiffness along the direction of insertion but high lateral and angular compliances K_x and K_θ, and it projects the center of compliance near the tip of the peg (hence the name remote center compliance) to overcome small lateral and angular errors in the peg's position and orientation in response to the contact forces applied to the peg by the hole during insertion (Fig. 36.18). The large lateral and angular compliance of the RCC also helps to avoid *wedging* and *jamming*, conditions that cause the peg to stick in two-point contacts due to contact forces in direct opposition (wedging) or ill-proportioned contact forces/moments (jamming). Because the RCC is low cost and reliable for fast insertions, it has long seen successful industrial applications.

However, the RCC has limitations. It only applies to round peg-in-hole insertions, and a particular RCC works better for a peg and hole of a particular size. Attempts have been made to expand the applicability of the RCC. Some attempts try to make certain param-

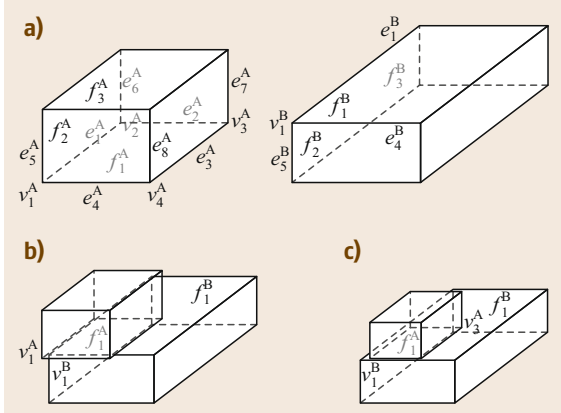


Fig.36.17a–c Indistinguishable contact states due to positional uncertainty of the top part; **(a)** two objects, **(b)** state 1, **(c)** state 2

eters of an RCC adjustable to achieve variable compliance or projection (i. e., the position of the remote center) [36.42–44]. A spatial RCC (SRCC) [36.45] was proposed to achieve square-peg-in-hole insertions. A notable difference with this extension is the combinatorial explosion of the number of possible contact states, given the range of the position and orientation uncertainty of the held part that have to be considered in the design of the device. Unlike the case of the round peg and hole, which is essentially a 2-D problem with a handful of different contact states, there are hundreds of possible different contact states between a square peg and a square hole. From each of these possible contact states, a motion leading the held part to the goal assembled state has to be realized by the device. This makes it difficult to design an RCC device that works for parts of more general and complex shapes with more possible contact states between them. This is a major limiting factor that prevent more extensions of the RCC to assembly of different parts.

Admittance Matrix

As an alternative method to the RCC, a particular form of manipulator force control, damping control, was proposed to achieve compliant motion of the held part by the manipulator and to correct small location errors of the held part during assembly. This approach eliminates the need to build a mechanical device like an RCC to achieve error correction. Among the force control laws [36.46], damping control is a common strategy, where a commanded velocity of the held part is modified based on the sensed force caused by contact between the held part and the environment. The resulted actual velocity leads to reduction and hopefully eventual correction of small position or orientation errors of the held part. Let $\boldsymbol{v}$ be a six-dimensional (6-D) vector representing the actual translational and angular velocity of the held part, $\boldsymbol{v}_0$ be the six-dimensional commanded velocity, and $\boldsymbol{f}$ be a six-dimensional vector representing the sensed force and moment. A linear damping control law is described as

$$\boldsymbol{v} = \boldsymbol{v}_0 + \mathbf{A}\boldsymbol{f} \,,$$

where $\mathbf{A}$ is a 6×6 matrix, called an *admittance matrix* or accommodation matrix.

The effectiveness of such a damping control law depends on the existence and finding of a proper admittance matrix $\mathbf{A}$. There is considerable research on the design of a single $\mathbf{A}$ that can make an assembly operation successful regardless of what contact states the held peg may encounter in the process [36.47–51]. This is aimed at cases where a single commanded velocity would be sufficient to achieve an assembly operation when there were no uncertainty or error, such as certain peg-in-hole insertion operations. One main approach to design $\mathbf{A}$ is based on explicit kinematic and static analysis of contact conditions under all possible contact states and mating requirements, which result in a set of linear inequalities as constraints on $\mathbf{A}$. Learning is also used [36.50] to obtain an $\mathbf{A}$ that minimizes the force $\boldsymbol{f}$ without causing instability. Another approach [36.51] applies perturbations to the end-effector during insertion in order to obtain richer force information.

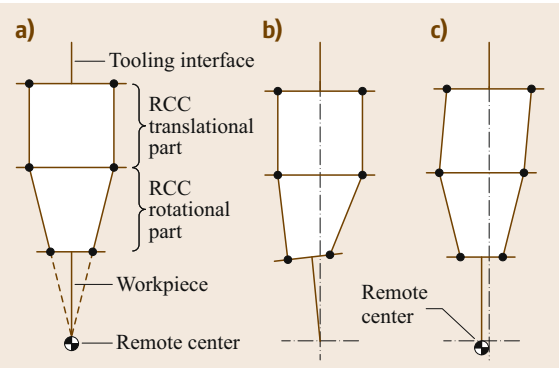


Fig. 36.18 (a) Planar representation of RCC. (b) Rotational part of RCC allowing workpiece to rotate. (c) Translational part of RCC allowing workpiece to translate

Learning Control for Assembly

Another category of approaches is to learn proper control for a particular assembly operation through stochastic or neural-network-based methods [36.52–56]. The essence of most of these approaches is to learn to map a reaction force upon the held object caused by contact to the next commanded velocity in order to reduce errors and to achieve an assembly operation successfully. A more recent approach [36.56] maps fused sensory data of pose and vision obtained during human demonstration of assembly tasks to compliant motion signals for successful assembly.

A different approach observes assembly tasks performed by human operators through vision [36.57] or in a virtual environment [36.58, 59] and generates a motion strategy necessary for the success of the task that consists of a sequence of recognized contact state transitions and associated motion parameters.

As a sequence of commanded velocities can be generated, unlike RCC or strategies based on a single admittance matrix described above, can be applied to cases with large uncertainties. However, the learned controllers are task dependent.

More recently, an approach combining vision and force sensing for assembly was introduced [36.60] based on measuring the deformation of parts through vision to adjust desired forces for mating and achieve

Avidbots Ex1016
Page 946 of 2154

force control through force sensing and an impedance controller. The approach was applied to the task of mating two power connectors. The effect was similar to having an RCC device on the end-effector.

All of the above assembly motion strategies do not require explicit recognition of contact states during their execution.

36.4.3 Active Compliant Motion

Active compliant motion is characterized by error correction based on online identification or recognition of contact states in addition to feedback of contact forces. The capability for active compliant motion allows a robot the flexibility to deal with a much broader range of assembly tasks with large uncertainties and tasks beyond assembly where compliance is required. An active compliant motion system generally requires the following components: a planner to plan compliant motion commands, an identifier to recognize contact states, state transitions, and other state information during task operation, and a controller to execute compliant motion plans based on both low-level feedback provided by sensors and high-level feedback provided by the identifier. Research on each component is described below.

Fine Motion Planning

Fine motion planning refers to planning fine-scale motions to make an assembly task successful in spite of significant uncertainties. A general approach was proposed [36.61] based on the concept of preimages in configuration space (*C*-space) [36.35] to devise motion strategies that would not fail in the presence of uncertainties. Given a goal state of the held part defined by a region of goal configurations, a *preimage* of the goal state encodes those configurations from which a commanded velocity will guarantee that the held part reaches the goal state recognizably in spite of location and velocity uncertainties (Fig. 36.19).

Given the initial location of the held part and the goal state, the preimage approach generates a motion plan in backward chaining by finding the preimage of the goal state associated with a commanded velocity and then the preimage of the preimage, and so on, until a preimage that includes the initial configuration of the held part is found. Figure 36.20 shows an example.

The sequence of commanded velocities associated with the sequence of preimages (starting from the initial preimage that includes the initial location) forms the motion plan that guarantees the success of the task. Starting from the initial preimage, each subsequent preimage in the sequence can be viewed as a subgoal state. In this approach, compliant motions are preferred wherever possible because they typically produce larger preimages [36.61] than pure positioning motions, and subgoal states are often contact states. The approach was further extended [36.62] to include modeling uncertainties of objects.

However, the computability of the approach is a major problem. The requirement of both goal reachability and recognizability under sensing uncertainties, which are intertwined issues, complicates the computation of preimages. It has been shown [36.63] that the time complexity of generating a plan can be double exponential in nmr, where n is the number of plan steps, m is the environment complexity (of the physical space), and r is the dimension of the configuration space. By separating reachability and recognizability and restricting the recognizability power of the original preimage model, computability was improved [36.64, 65].

As an alternative, a two-phase approach is to simplify the fine motion planning into (1) global and offline nominal path planning assuming no uncertainty and (2) local and online replanning to deal with unintended contacts due to uncertainties. Different variations of the two-phase approach have been proposed [36.66–72]. The success of such an approach depends on successful online identification of contact states (Sect. 36.4.3).

Several researchers also studied the representation and propagation of uncertainties and constraints that have to be satisfied for the success of a task [36.73–77].

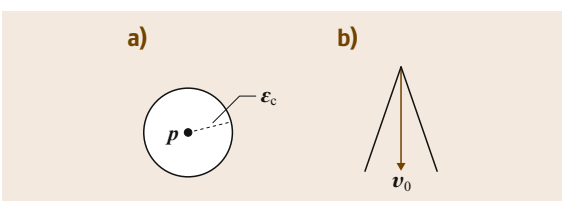


Fig.36.19a,b Location and velocity uncertainties (in the *C*-space). **(a)** The actual configuration of a part could be inside an uncertainty ball centered at an observed configuration. **(b)** The actual velocity of a part could be inside an uncertainty cone of a commanded velocity

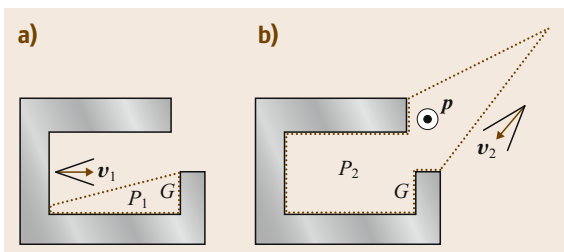


Fig.36.20a,b Backward chaining of preimages: P_1 is the preimage of the goal region G **(a)**, and P_2 is the preimage of P_1 **(b)**

Compliant Motion Planning

Compliant motion planning focuses on planning motions of objects always in contact. *Hopcroft* and *Wilfong* [36.78] proved that, if two arbitrary objects in contact can be moved to another configuration where they are also in contact, then there is always a path of contact configurations (on the boundary of configuration space obstacles, or C-obstacles) connecting the first and second contact configurations. Therefore, not only is compliant motion desirable in many cases, but it is always possible given initial and goal contact configurations.

Because compliant motion occurs on the boundary of C-obstacles, planning compliant motion poses special challenges not present in collision-free motion planning: it requires the exact information of contact configurations on the boundary of C-obstacles. Unfortunately computing C-obstacles exactly remains a formidable task to date. While there are exact descriptions of C-obstacles for polygons [36.79, 80], which are three dimensional, there are only approximations of C-obstacles for polyhedra [36.81, 82], which are six dimensional. If m and n indicate the complexity of two polyhedral objects in contact, the complexity of the corresponding C-obstacle is $\Theta(m^6n^6)$ [36.83].

Researchers have focused on either reducing the dimensionality and scope of the problem or getting around the problem of computing C-obstacles altogether. Some researchers have studied compliant motion planning on C-obstacles of reduced dimensions [36.72, 84, 85]. It is more popular to plan compliant motions based on a predetermined graph of topological contact states to avoid the problem of computing C-obstacles [36.86–89]. In particular, one approach [36.87, 88] models an assembly task of polygonal parts as a discrete event system using Petri nets. However, contact states and transitions are often generated manually, which is awfully tedious for even assembly tasks of simple geometry [36.45] and is practically infeasible for complex tasks due to the huge number of different contact states.

Therefore, automatic generation of a contact state graph is desirable and even necessary. A method was first developed to enumerate all possible contact states and their connections between two convex polyhedral objects [36.90]. Recall that a contact state between two convex polyhedra can be described by a single principal contact (Sect. 36.4.1), and any principal contact between two topological surface elements of two convex polyhedra describes a geometrically valid contact state. These are nice properties and greatly simplify the problem of contact state graph generation. In general, however, to construct a contact state graph between two objects automatically requires the handling of two rather difficult issues:

1. How to generate *valid* contact states, i. e., how to tell if a set of principal contacts corresponds to a geometrically valid contact state, given the geometries of objects, and
2. How to link one valid contact state to another in the graph, i. e., how to find the neighboring (or adjacency) relations among the regions of contact configurations belonging to different contact states in the contact configuration space.

A general and efficient divide-and-merge approach was introduced [36.38] for automatically generating a contact state graph between two arbitrary polyhedra. Each node in the graph denotes a contact state, described by a topological *contact formation* (CF) [36.37] as a set of principal contacts and a configuration satisfying the CF. Each edge connects the nodes of two neighboring contact states. Figure 36.21 shows an example contact state graph between two planar parts.

The approach handled the above two issues simultaneously by directly exploiting both topological and geometrical knowledge of contacts in the *physical space* of objects and by dividing the problem into simpler subproblems of generating and merging special subgraphs. Specifically, the approach takes advantage of the fact that a contact state graph can be divided into special subgraphs called the *goal-contact relaxation* (GCR) graphs, where each GCR graph is defined by a locally most constrained valid contact state, called the seed, and its less-constrained neighboring valid contact states, which is easier to generate because of several properties. The main properties include:

- Given a valid contact state CS_i all of its less constrained neighboring contact states can be hy-

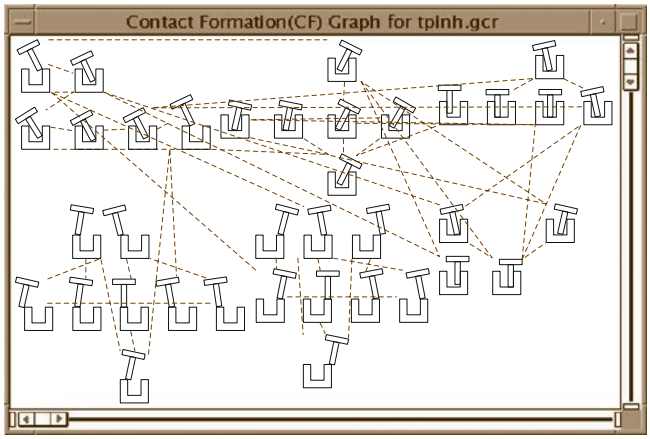


Fig. 36.21 A contact state graph between two planar parts

pothesized topologically from the principal contacts in CS_i.

- A hypothesized less constrained neighboring contact state CS_j is valid, if and only if there exists a compliant motion to relax certain constraints of CS_i to obtain CS_j which does not result in any other contact state, called neighboring relaxation.
- Neighboring relaxation can often be achieved by instantaneous compliant motion.

With this approach, a contact state graph of several hundreds or thousands of nodes and links can be generated in a few seconds.

With a contact state graph, the problem of compliant motion planning can be decomposed into two simpler subproblems at two levels: (1) high level: graph search for state transitions from one node to another in a contact state graph, and (2) low level: contact motion planning within the set of contact configurations constrained by the *same* contact state (and the same contact formation), called *CF-compliant motion planning*. A general contact motion plan crossing several contact states can be considered as consisting of segments of CF-compliant motions in different contact states. One approach plans [36.91] CF-compliant motions based on random sampling of CF-compliant configurations and extending the probabilistic roadmap motion planning technique [36.92].

Contact State Identification

Successful execution of a fine-motion or compliant motion plan depends on correct online identification of contact states during execution. Contact state identification uses both model information of contact states (including topological, geometrical and physical information) and sensory information of location and force/torque of the end-effector. The presence of sensing uncertainties make the identification problem nontrivial. Approaches for contact state identification are different in ways of dealing with uncertainties.

One class of approaches obtain the mapping between sensory data (in the presence of sensing uncertainties) and corresponding contact states through learning. Models for learning include hidden Markov models [36.93, 94], thresholds [36.95, 96], neural network structures, and fuzzy classifiers [36.97–101]. Training data are obtained either in an unsupervised way or by human task demonstration. Such approaches are task dependent: a new task or environment requires new training.

Another class of approaches are based on analytical models for contact states. A common strategy is to predetermine the set of configurations, constraints on configurations, or the set of forces/torques or force/torque constraints that are possible for each contact state and match such information against the sensed data to identify contact states online. Some approaches do not consider the effect of uncertainties [36.102, 103], while others consider uncertainties modeled either by uncertainty bounds or by probability distributions [36.36, 89, 104–108]. An alternative strategy is based on checking the distance between contacting elements in Cartesian space: either by growing the objects with pose uncertainty and intersecting the obtained regions [36.109], or by combining the distance between the objects with several other factors such as the instantaneous approach direction [36.110].

In terms of how sensory data are used, some identification schemes are rather static because they do not consider the prior motion and state identifications before the current contact state is reached, while others use prior history or even use new motion or *active sensing* to help current identification. The latter include approaches that perform simultaneous contact state identification and parameter estimation. In such approaches, an analytical contact state model is expressed as a function of uncertain parameters, such as the pose or dimension of an object in contact. At each contact state, some of the uncertain parameters can be observed or estimated with reduced uncertainty. For example, if the held object is at a face–face contact with another object, parameters describing the normals of the contacting faces can be estimated with the help of force/torque sensing. During a task execution (i. e., the execution of a motion plan), the uncertain parameters are estimated along with the identification of contact states with increasing accuracy, which in turn makes the subsequent contact state identifications more accurate. This also improves force control and contact state transition monitoring. The increased performance, however, comes at a higher computational cost.

Simultaneous contact state identification and parameter estimation is most often done by ruling out contact state models that are inconsistent with sensed data or parameter estimation results [36.111–117]. One notable exception [36.118] performs a truly simultaneous estimation by describing the system as a hybrid joint probability distribution of contact states and parameters.

Active sensing is about deliberate use of applied force/torque or motion to better aid contact state identification and parameter estimation. Earlier research focused on simple force strategy [36.119] or movability tests [36.36, 110]. More recently, methods were introduced [36.120] to design sequences of contact state transitions and compliant motion strategies that were optimized for active sensing to determine all uncertain

geometric parameters while achieving certain goal contact state.

From a different perspective, an approach was introduced to analyze contact state distinguishability and unknown/uncertain parameter identifiability [36.121] during the design phase of a contact state identifier or parameter estimator.

36.4.4 Compliant Motion Under Manipulator Constraints

A general compliant motion plan (as the output of a compliant motion or fine motion planner) usually consists of a sequence of contact states, and within each contact state, a path of contact configurations compliant to the contact state and leads to the next contact state in the sequence. Each contact configuration is described in terms of the parts in contact. However, it takes a robot manipulator to move one part against the others, especially in assembly tasks. Therefore, whether a contact state can be formed and whether a compliant motion and a contact state transition is possible is subject to the constraints of the manipulator.

Feasibility of Contact States and Compliant Motions

A contact state graph should be re-visited by considering manipulator constraints [36.122]. Contact configurations compliant to a given contact state are sampled and applied inverse kinematics to check if there is a feasible manipulator joint-space configuration to realize the contact state. If no feasible manipulator configuration can be found after a number of samples are considered, the contact state is considered at least very difficult to reach, and is discarded from the graph as an infeasible state.

Given two feasible adjacent contact states, a compliant motion for their transition should also be checked for feasibility under manipulator constraints. The main issue here is how to deal with possible singularities along the compliant path (of contact configurations). Certain singularities can be escaped if the manipulator is redundant (i. e., with more than six degrees of freedom) [36.123]. With inescapable singularities, the compliant path of the moved object is not feasible and has to be either altered to avoid singularity [36.124] or abandoned. However, an altered path may no longer be compliant to the desired contact states. In practice, it is preferred to check the feasibility of a compliant path starting from the goal contact state of an assembly task backward to make sure that the feasible portion leading to the goal as well as the feasible contact states encountered are discovered and preserved [36.125].

Execution of Feasible Compliant Motion Plans

Depending on whether the manipulator is equipped with a force/torque sensor or not, different control schemes can be considered for executing a compliant motion plan that includes multiple contact state transitions (VIDEO 356).

If the manipulator has a force/torque sensor, a hybrid position/force controller can be used. It requires a specification that separates dimensions of force control from those of position/velocity control. Such specifications not only have to vary for different contact states but may have to vary from one contact configuration to another within the same contact state, i. e., a control specification is a function of time or configuration along a compliant trajectory in general. Moreover, it is far from trivial to make control specifications for complex contact states involving multiple principal contacts [36.126].

An approach was introduced [36.127] to convert automatically a compliant path of contact configurations across a sequence of contact states into wrench, twist, and position control signals $\boldsymbol{w}(t)$, $\boldsymbol{t}(t)$, and $\boldsymbol{p}(t)$ for a hybrid controller to execute the plan. The approach was experimentally verified successfully.

More recently, an approach was introduced [36.128] to achieve force control by estimating contact force/torque from the joint torques without using a force sensor for the end-effector force/torque or torque sensors for individual joints. By reducing the integral gain of the joint PID (proportional–integral–derivative) controller, the approach acts like a high-pass filter to detect sharp force/torque changes more effectively, which often indicates contact state changes. Combining such detection with position control, the approach was able to execute an assembly motion plan successfully, even though compliant motion could not always be maintained with position control.

Other approaches to realize force control without a force sensor requires the dynamic model of the manipulator [36.128–130].

However, there is not yet a full integration of planning, online contact state identification, online replanning (to deal with contact states that are off the preplanned path due to uncertainty), and compliant motion control.

36.5 Unifying Feedback Control and Planning

In previous sections we discussed task-level control methods and planning algorithms. Both task-level control and planning methods are able to address specific constraints imposed on robot motion in the context of manipulation tasks. However, as illustrated in Fig. 36.3, neither of these categories of methods is able to address all constraints completely. Control remains susceptible to local minima and thus cannot guarantee that a particular motion will lead to the desired result. Planning methods, on the other hand, overcome the problem of local minima by projecting possible motions into the future to predict if a sequence of motion will lead to success. Given certain assumptions, the resulting plan can be guaranteed to succeed. However, the computations associated with this process are generally too computationally complex to satisfy the feedback requirements of manipulation tasks. Unfortunately, this means that control and motion planning by themselves are unable to address the problem of moving robots for general manipulations tasks.

In this final section of this chapter, we will review efforts to combine the advantages of control methods and planning methods into a single approach to determine the motion of a robot. These efforts aim to create methods that avoid the susceptibility to local minima while satisfying the feedback requirements. Early attempts of integrating planning and control occurred in the early 1990s. Not all of these efforts are specifically directed at manipulation but many contain insights relevant to the topic. In this section, we will review these efforts and also discuss some of the more recent research efforts aimed at unifying motion planning and feedback control.

Motion planning (Chap. 7) and motion control (Chap. 8) have traditionally been regarded as two distinct areas of research. However, these areas have many features in common. Both areas are concerned with the motion of robotic mechanisms, and more importantly planning and control methods both determine a representation that maps robot state to robot motion. In the case of feedback control, this representation is a potential function defined for a given region of the state space. The gradient of the potential function at a specific state encodes the motion command. In contrast, motion planning determines motion plans. These motion plans also encode motions for a set of states.

Motion planning and feedback control also differ in certain aspects. A motion planner generally makes stronger assumptions than a controller about the environment, the ability to assess its state, and the changes that can occur in the environment. Motion planning also requires the ability to project the robot's state into the future, given its current state and a particular action. By using this ability together with global information about the environment, motion planners can determine motions that are not susceptible to local minima. This positive characteristic of motion planners, however, results in significantly increased computational cost (Chap. 7). The large discrepancy in computational requirements for planning and control and the resulting divergence of computational techniques may explain the current separation of the two fields. Researchers are beginning to attempt to reverse this separation by devising a unified theory of planning and control.

36.5.1 Feedback Motion Planning

Feedback motion planning combines planning and feedback into a single motion strategy. A planner considers global information to compute a feedback motion plan free of local minima. Such a feedback motion plan can be interpreted as a potential function or vector field whose gradient will lead the robot to the goal state from any reachable part of the state space [36.18]. Local minima-free potential functions are also called navigation functions [36.131, 132]. Given the current state of the robot and the global navigation function, feedback control is used to determine the robot's motion. The consideration of feedback about the robot's state in the context of a global navigation function reduces the susceptibility to sensing and actuation uncertainty.

Feedback motion planning, in principle, can address the entire spectrum of motion constraints and their feedback requirements (Fig. 36.3). Given a global navigation function that considers all motion constraints, the feedback requirements can easily be satisfied, since the feedback motion plan already specifies the desired motion command for the entire state space. Obviously, the major challenge in feedback motion planning is the computation of such a navigation function or feedback motion plan. The problem becomes particularly difficult in the context of a manipulation task, since the state space (or configuration space) changes each time the robot grasps or releases an object in the environment (Sect. 36.3). This change makes it necessary to recompute the feedback motion plan. Frequent recomputation is also necessary in dynamic environments, where the motion of obstacles can repeatedly invalidate a previously computed feedback motion plan.

In the remainder of this section, we will review a variety of methods for computing navigation functions. In general, the problem of efficiently computing feedback motion plans for manipulation tasks remains unsolved. We therefore also present methods that do not explicitly

consider manipulation. These methods can be divided into three categories: (1) exact methods, (2) approximate methods based on dynamic programming, and (3) approximate methods based on composing and sequencing simpler potential functions.

The earliest exact methods for the computation of navigation functions are applicable to simple environments with obstacles of specific shapes [36.131–133]. Approximate methods based on discretized spaces (grids) overcome this limitation but possess an exponential computational complexity in the number of dimensions of the state space. These approximate navigation functions are called numerical navigation functions [36.134]. These navigation functions are often used for applications in mobile robotics. Due to the low-dimensional configuration space associated with mobile robots, they can generally solve motion planning problems robustly and efficiently.

Some physical processes, such as heat transfer or fluid flow, can be described by a specific type of differential equation, called harmonic functions. These functions possess properties that make them suitable as navigation functions [36.135–139]. Navigation functions based on harmonic functions are most commonly computed in an approximate, iterative fashion. The requirement for iterative computation increases the computational cost relative to the simpler numerical navigation functions [36.134].

More recent methods for the computation of numerical navigation functions consider differential motion constraints at a significantly reduced computational cost [36.140]. These methods rely on classical numerical dynamic programming techniques and numerical optimal control. However, in spite of their reduced computational complexity, these methods remain too computationally costly to be applied to manipulation tasks with many degrees of freedom in dynamic environments.

Navigation functions can also be computed by composing local potential functions based on global information. This is illustrated in Fig. 36.22. The goal is to compute a navigation function for the entire configuration space C. This is accomplished by sequencing overlapping funnels. Each of the funnels represents a simple, local potential function. By following the gradient of this potential function, motion commands can be determined for a subset of the configuration space. If the funnels are sequenced correctly, the composition of local funnels can yield a global feedback plan. This feedback plan can be viewed as a hybrid system [36.141] in which the sequencing of funnels represents a discrete transition structure, whereas the individual controllers operate in a continuous domain. The composition of funnels is considered planning. The consideration of global information during the planning permits to determine a funnel composition that avoids local minima.

One of the earliest methods based on the global composition of local funnels was proposed by *Choi* and *Latombe* [36.142]. In this method, the state space of the robot is decomposed into convex regions. The connectivity of these regions is analyzed to determine global information about the state space. This information can be used to combine simple, local potential functions, one for each convex regions of state space, into a local minima-free potential function. More rigorous approaches based on this idea have been developed. These approaches can consider the dynamics of the robot and nonholonomic motion constraints [36.143–145]. These methods have only been applied to low-dimensional state spaces and cannot easily be applied to manipulation tasks.

The random neighborhood graph [36.146] is a sampling-based method of computing a decomposition of the overall state space. As before, a navigation function can be computed by analyzing the global connectivity of the decomposition and imposing adequate local potential functions for each subdivision. A specialized method following the same principle for planar robots in polygonal environments has also been proposed [36.147]. The idea of composing local potential functions has also been applied successfully to a complex robot control task [36.148]. Finally, in [36.149] a general and efficient method of computing a smooth feedback plan over a cylindrical algebraic decomposition of configuration space has been proposed.

The computation of a navigation function over the entire configuration space quickly becomes intractable as the dimensionality of the space increases. To overcome this challenge, in particular in the context of autonomous mobile manipulation, workspace heuristics have been employed to determine a navigation function efficiently. This navigation function does not cover the entire configuration space but only those regions heuristically determined to be relevant to the motion problem [36.150]. This method of generating feedback

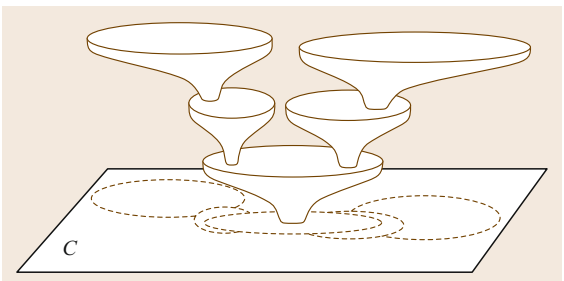


Fig. 36.22 Composing funnels into a navigation function

plans is able to satisfy the various motion constraints depicted in Fig. 36.3 and their respective feedback requirements.

36.5.2 Augmenting Global Plans with Feedback

The manipulation planning techniques described in Sect. 36.3 are not susceptible to local minima, as they consider global state space information. Due to the computational complexity associated with the consideration of global information, these planning techniques are not able to satisfy the feedback requirements of manipulation tasks. However, the required frequency of feedback about global motion are relatively low: the global connectivity of the configuration space changes relatively infrequently (Fig. 36.3). It would therefore be possible to consider feedback for global motion at the slow rates the planner can accommodate, while considering feedback for other motion constraints at higher frequencies. To achieve this, global motion plans have to be augmented with reactive components that incrementally modify the global plan in response to feedback from the environment. As long as the global connectivity information captured by the plan remains valid, the incremental modifications can ensure that all other motion constraints, ranging from task requirements to reactive obstacles avoidance, are satisfied.

The elastic-band framework [36.151] augments global plans with reactive obstacle avoidance. A global configuration space path, determined by a planner, is covered with local potential functions, each of which is derived from the local distribution of obstacles around the path. These local potentials cause the path to deform so as to maintain a minimum distance from obstacles. Visually, the path behaves as an elastic band that is deformed by the motion of obstacles. The local potential functions, together with the global path, can be viewed as a navigation function for a local region of the configuration space. Integrated with a global planner and replanner, the elastic-band framework permits real-time obstacle avoidance that is not susceptible to local minima. However, the feedback frequency for global motion remains limited by the global motion planner. Specific task constraints have not been integrated into the elastic-band framework; consequently, its application to manipulation tasks is limited.

In its original formulation, the elastic-band framework assumed that all degrees of freedom of the robot are holonomic. An extended formulation augments motion paths for nonholonomic platforms with reactive components [36.152].

The elastic-strip framework [36.153] also augments global motion plans with reactive obstacle avoidance. In addition to reactive obstacle avoidance, however, the elastic-strip framework can accommodate task constraints. Similarly to an elastic band, an elastic strip covers a global path with local potential functions. In contrast to the elastic-band framework, these potential functions are based on task-level controllers (Sect. 36.2) and therefore allow the task-consistent modification of the global path. The elastic-strip framework is therefore well suited for the execution of manipulation plans in dynamic environments. An elastic strip will be incrementally modified to represent a constraint-consistent trajectory, as long as the global information captured by the underlying plan remains valid. The elastic-strip framework has been applied to a variety of manipulation tasks on a mobile manipulation platform.

Extending the elastic-band and elastic-strips frameworks, the elastic-roadmap framework combines reactive task-level control with efficient global motion planning [36.150]. The elastic roadmap represents a hybrid system of task-level controllers that are composed into a navigation function, thereby satisfying the motion constraints depicted in Fig. 36.3 and their respective feedback requirements.

36.6 Conclusions and Further Reading

In this chapter, an overview of motion generation and control strategies in the context of robotic manipulation tasks has been provided. Issues related to modeling the interfaces between the robot and the environment at the different time scales of motion and incorporating sensing and feedback were considered. Manipulation planning was introduced as an extension to the basic motion planning problem, which can be modeled as a hybrid system of continuous configuration spaces arising from the act of grasping and moving parts in the environment. The important example of assembly motion has been discussed through the analysis of contact states and compliant motion control. The operational space framework, as described in Sect. 36.2, permits the position and force control of operational points on a serial-chain manipulator. A number of extensions have been proposed in the literature, extending the framework to situations with multiple concurrent contact points, cooperative manipulation scenarios, and branching kinematic chains. For manipulation plan-

ning, extensions involving grasp and regrasp planning, multiple robots, multiple parts, and movable obstacles have been considered. In this section we will briefly discuss these extensions and refer to the appropriate literature.

36.6.1 General Contact Models

The hybrid force/motion control described by *Raibert* and *Craig* [36.154] has been shown to have shortcomings [36.155]. A general contact model for dynamically decoupled force/motion control in the context of operational space control that overcomes these problems has been presented in [36.155]. This general contact model has been extended into a compliant motion control framework to enable force control for multiple contact points [36.126] in nonrigid environments, and to enable force control for multiple contact points on different links of a kinematic chain [36.156]. The operational space framework for task-level control has been applied for the real-time simulation of complex dynamic environments [36.157]. Contact points between objects in the environment are modeled as operation points, resulting in a mathematically elegant framework for resolving impulses and contact constraints for moving bodies.

36.6.2 Cooperative Manipulation Control

If multiple task-level controlled robots collaborate to manipulate an object, they form a closed kinematic chain that is connected through the object. The dynamics of the entire system can be described using the notion of an augmented object [36.158, 159], in which the dynamics of the manipulators and the object are combined to form a model of the overall system. The internal forces that occur at the grasp points during motion can be modeled using the virtual linkage framework [36.160]. A number of alternative strategies for the cooperative manipulation of multiple robots outside of the operational space framework have been proposed [36.161–168].

36.6.3 Control of Branching Mechanisms

So far, we have implicitly assumed that the task-controlled robot consists of a single kinematic chain. This assumption does not hold in the case of kinematically more complex mechanisms, such as humanoid robots (Chap. 67). These robots can consist of multiple branching kinematic chains. If we consider the torso of a humanoid robot to be the robot's base, for example, then the legs, arms, and the head represent five kinematic chains attached to this base. We call such a mechanism a branching mechanism if it does not contain any closed kinematic loops. Tasks performed by a branching mechanism may require the specification of operational points on any one of those branches; for example, while the legs perform locomotion the hands achieve a manipulation task and the head is oriented to maintain visibility of the manipulated object.

The operational space framework has been extended to task-level control of branching mechanisms [36.169]. This extension combined operational points and associated Jacobians and computes an operational space inertia matrix that combines all operational points. This matrix can be computed efficiently with an algorithm that in practice is linear in the number of operational points [36.170, 171]. More recently, the operational space framework has been applied to quadrupedal locomotion [36.172].

36.6.4 Nonholonomic Mobile Manipulation

All previously discussed work on task-level control assumes that the degrees of freedom are holonomic (Chap. 40). For manipulator arms this is generally a valid assumption. The most common type of mobility platform, however, is based on either differential drives, synchro-drives, or Ackerman steerings (point to a section), all of which are subject to nonholonomic constraints. The operational space framework has been extended to mobile manipulation platforms that combine a nonholonomic mobility platform with a holonomic manipulator arm [36.173–175], enabling the task-level control of a large class of mobile manipulation platforms.

36.6.5 Learning Models with Uncertainty

The efficacy of operational space control depends on the accuracy of the dynamic model of the robot. In particular when multiple behaviors are executed using null-space projections, modeling errors can have significant effects. To overcome the reliance on accurate dynamic models, reinforcement learning can be used to learn operational space controllers [36.176].

36.6.6 Grasping and Regrasp Planning

Deciding the intermediate stable configurations for a part necessary to complete a manipulation task and selecting proper grasps are both difficult problems in themselves that operate over a continuum of possible solutions. *Simeon* et al. developed a manipulation planning framework that considers continuous grasps and placements [36.177]. However, selecting from among all possible grasp configurations and deciding when

to regrasp is still an open research problem. For an overview of various grasp quality metrics, see *Miller* and *Allen* [36.178, 179] and Chap. 38.

36.6.7 Multiple Parts

The manipulation planning framework nicely generalizes to multiple parts, $\mathcal{P}_1, \ldots, \mathcal{P}_k$. Each part has its own C-space, and C is formed by taking the Cartesian product of all the part C-spaces with the manipulator C-space. The set C_{adm} is defined in a similar way, but now part–part collisions also have to be removed, in addition to part–manipulator, manipulator–obstacle, and part–obstacle collisions. The definition of C_{stable} requires that all parts be in stable configurations; the parts may even be allowed to stack on top of each other. The definition of C_{grasp} requires that one part is grasped and all other parts are stable. There are still two modes, depending on whether the manipulator is grasping a part. Once again, transitions occur only when the robot is in $C_{\text{trans}} = C_{\text{stable}} \cap C_{\text{grasp}}$.

36.6.8 Planning for Multiple Robots

Generalizing to k robots would lead to $2k$ modes, in which each mode indicates whether each robot is grasping a part. Multiple robots may even be allowed to grasp the same part, which leads to the interesting problem of planning for closed kinematic chains and cooperative motion (Chap. 39). *Koga* addressed multi-arm manipulation planning where the same object must be grasped and moved by several manipulator arms [36.21]. Another generalization could allow a single robot to grasp more than one part simultaneously.

36.6.9 Planning for Closed Kinematic Chains

The subspace of C that results from maintaining kinematic closure arises when multiple robots grasp the same part, or even when multiple fingers of the same hand grasp a single part (Chap. 19). Planning in this context requires that paths remain on a lower-dimensional variety for which a parameterization is not available. Planning the motion of parallel mechanisms or other systems with loops typically requires maintaining multiple closure constraints simultaneously (Chap. 18).

36.6.10 Planning with Movable Obstacles

In some cases, the robot may be allowed to reposition obstacles in the environment rather than simply avoid them. *Wilfong* first addressed motion planning in the presence of movable obstacles [36.180]. Wilfong proved that the problem is PSPACE-hard even in two-dimensional environments where the final positions of all movable objects are specified. *Erdmann* and *Lozano-Perez* considered coordinated planning for multiple moving objects [36.181]. *Alami* et al. presented a general algorithm for the case of one robot and one movable object and formulated the space of grasping configurations into a finite number of cells [36.17]. *Chen* and *Hwang* developed a planner for a circular robot that is allowed to push objects aside as it moves [36.182]. *Stilman* and *Kuffner* considered movable obstacles in the context of navigation planning [36.183], and manipulation planning [36.184]. *Nieuwenhuisen* et al. also developed a general framework for planning with movable obstacles [36.185].

36.6.11 Nonprehensile Manipulation

Lynch and *Mason* explored scenarios in which grasping operations are replaced by pushing operations [36.186]. The space of stable pushing directions imposes nonholonomic constraints on the motion of the robot, which opens up issues related to controllability. Related issues also arise in the context of part-feeders and manipulation for manufacturing and assembly (Chap. 54).

36.6.12 Assembly Motion Extensions

The work in assembly motion described in this chapter has focused on rigid-part assembly, and the parts considered are mostly polyhedral or of simple nonpolyhedral shapes, such as round pegs and holes. More recent work on contact state analysis for curved objects can be found in [36.187]. An emerging field in assembly is micro/nanoassembly ( ; Chap. 27). Flexible or deformable part assembly has also begun to attract attention [36.188–190]. Virtual assembly [36.191] for simulation and prototyping is another interesting area of study.

Video-References

VIDEO 356 Reducing uncertainty in robotics surface assembly tasks
available from http://handbookofrobotics.org/view-chapter/36/videodetails/356

VIDEO 357 Autonomous continuum grasping
available from http://handbookofrobotics.org/view-chapter/36/videodetails/357

VIDEO 358 Robotic assembly of emergency stop buttons
available from http://handbookofrobotics.org/view-chapter/36/videodetails/358

VIDEO 359 Meso-scale manipulation: System, modeling, planning and control
available from http://handbookofrobotics.org/view-chapter/36/videodetails/359

VIDEO 360 Grasp and multi-fingers-3 cylindrical-peg-in-hole demonstration using manipulation primitives
available from http://handbookofrobotics.org/view-chapter/36/videodetails/360

VIDEO 361 Demonstration of multi-sensor integration in industrial manipulation
available from http://handbookofrobotics.org/view-chapter/36/videodetails/361

VIDEO 363 Control pre-imaging for multifingered grasp synthesis
available from http://handbookofrobotics.org/view-chapter/36/videodetails/363

VIDEO 364 Robust and fast manipulation of objects with multi-fingered hands
available from http://handbookofrobotics.org/view-chapter/36/videodetails/364

VIDEO 366 Whole quadruped manipulation
available from http://handbookofrobotics.org/view-chapter/36/videodetails/366

VIDEO 367 The mobipulator
available from http://handbookofrobotics.org/view-chapter/36/videodetails/367

VIDEO 368 Handling of a single object by multiple mobile robots based on caster-like dynamics
available from http://handbookofrobotics.org/view-chapter/36/videodetails/368

VIDEO 369 Rollin' Justin – Mobile platform with variable base
available from http://handbookofrobotics.org/view-chapter/36/videodetails/369

VIDEO 370 Learning to place new objects
available from http://handbookofrobotics.org/view-chapter/36/videodetails/370

References

36.1 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)

36.2 H. Inoue: *Force Feedback in Precise Assembly Tasks*, Tech. Rep. Vol. 308 (Artificial Intelligence Laboratory, MIT, Cambridge 1974)

36.3 T. Lozano-Pérez, M. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robots, Int. J. Robotics Res. **31**(1), 3–24 (1984)

36.4 O. Khatib: A unified approach to motion and force control of robot manipulators: The operational space formulation, Int. J. Robotics Autom. **3**(1), 43–53 (1987)

36.5 O. Khatib, K. Yokoi, O. Brock, K.-S. Chang, A. Casal: Robots in human environments, Arch. Control Sci. **11**(3/4), 123–138 (2001)

36.6 G. Strang: *Linear Algegra and Its Applications* (Brooks Cole, New York 1988)

36.7 H. Seraji: An on-line approach to coordinated mobility and manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta, Vol. 1 (1993) pp. 28–35

36.8 J.M. Cameron, D.C. MacKenzie, K.R. Ward, R.C. Arkin, W.J. Book: Reactive control for mobile manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta, Vol. 3 (1993) pp. 228–235

36.9 M. Egerstedt, X. Hu: Coordinated trajectory following for mobile manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000)

36.10 P. Ögren, M. Egerstedt, X. Hu: Reactive mobile manipulation using dyanmic trajectory tracking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francicso (2000) pp. 3473–3478

36.11 J. Tan, N. Xi, Y. Wang: Integrated task planning and control for mobile manipulators, Int. J. Robotics Res. **22**(5), 337–354 (2003)

36.12 Y. Yamamoto, X. Yun: Unified analysis on mobility and manipulability of mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 1200–1206

36.13 L. Sentis, O. Khatib: Control of free-floating humanoid robots through task prioritization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005)

36.14 L. Sentis, O. Khatib: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives, Int. J. Hum. Robotics **2**(4), 505–518 (2005)

36.15 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robotics Res. **5**(1), 90–98 (1986)

36.16 M. Huber, R.A. Grupen: A feedback control structure for on-line learning tasks, Robotics Auton. Syst. **22**(3-4), 303–315 (1997)

36.17 R. Alami, J.P. Laumond, T. Siméon: Two manipulation planning algorithms, Proc. Workshop Algorithm. Found. Robotics (1994)

36.18 S.M. LaValle: *Planning Algorithms* (Cambridge Univ., Cambridge 2006)

36.19 R. Grossman, A. Nerode, A. Ravn, H. Rischel (Eds.): *Hybrid Systems* (Springer, Berlin, Heidelberg 1993)

Avidbots Ex1016
Page 956 of 2154

36.20 J.M. Ahuactzin, K. Gupta, E. Mazer: Manipulation planning for redundant robots: A practical approach, Int. J. Robotics Res. **17**(7), 731–747 (1998)

36.21 Y. Koga: On Computing Multi-Arm Manipulation Trajectories, Ph.D. Thesis (Stanford University, Stanford 1995)

36.22 D. Bertram, J.J. Kuffner, T. Asfour, R. Dillman: A unified approach to inverse kinematics and path planning for redundant manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1874–1879

36.23 D. Trivedi, C.D. Rahn, W.M. Kier, I.D. Walker: Soft robotics: Biological inspiration, state of the art, and future research, Appl. Bionics Biomech. **5**(3), 99–117 (2008)

36.24 B.A. Jones, I.D. Walker: Kinematics for multisection continuum robots, IEEE Trans. Robotics **22**(1), 43–55 (2006)

36.25 S. Neppalli, M.A. Csencsits, B.A. Jones, I. Walker: A geometrical approach to inverse kinematics for continuum manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3565–3570

36.26 J. Li, J. Xiao: Determining *grasping* configurations for a spatial continuum manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4207–4214

36.27 J. Xiao, R. Vatcha: Real-time adaptive motion planning for a continuum manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 5919–5926

36.28 J. Li, J. Xiao: Progressive generation of force-closure grasps for an n-section continuum manipulator, Proc. IEEE Int. Conf. Robotics Auto. (ICRA) (2013) pp. 4016–4022

36.29 J. Li, J. Xiao: Progressive, continuum grasping in cluttered space, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 4563–4568

36.30 G.S. Chirikjian, J.W. Burdick: Kinematically optimal hyperredundant manipulator configurations, IEEE Trans. Robotics Autom. **11**(6), 794–798 (1995)

36.31 G. Boothroyd: *Assembly Automation and Product Design*, 2nd edn. (Taylor Francis, Boca Raton 2005)

36.32 D. Whitney, O.L. Gilbert, M. Jastrzebski: Representation of geometric variations using matrix transforms for statistical tolerance analysis in assemblies, Res. Eng. Des. **64**, 191–210 (1994)

36.33 P. Jimenez: Survey on assembly sequencing: A combinatorial and geometrical perspective, J. Intell. Manuf. **I24**, 235–250 (2013)

36.34 B. Shirinzadeh: Issues in the design of the reconfigurable fixture modules for robotic assembly, J. Manuf. Syst. **121**, 1–14 (1993)

36.35 T. Lozano-Pérez: Spatial planning: A configuration space approach, IEEE Trans. Comput. **C-32**(2), 108–120 (1983)

36.36 R. Desai: On Fine Motion in Mechanical Assembly in Presence of Uncertainty, Ph.D. Thesis (University of Michigan, Ann Arbor 1989)

36.37 J. Xiao: Automatic determination of topological contacts in the presence of sensing uncertainties, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 65–70

36.38 J. Xiao, X. Ji: On automatic generation of high-level contact state space, Int. J. Robotics Res. **20**(7), 584–606 (2001)

36.39 S.N. Simunovic: Force information in assembly processes, Proc. 5th Int. Symp. Ind. Robots (1975) pp. 415–431

36.40 S.H. Drake: Using Compliance in Lieu of Sensory Feedback for Automatic Assembly, Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1989)

36.41 D.E. Whitney: Quasi-static assembly of compliantly supported rigid parts, ASME J. Dyn. Syst. Meas. Control **104**, 65–77 (1982)

36.42 R.L. Hollis: A six-degree-of-freedom magnetically levitated variable compliance fine-motion wrist: Design, modeling, and control, IEEE Trans. Robotics Autom. **7**(3), 320–332 (1991)

36.43 S. Joo, F. Miyazaki: Development of variable RCC and ITS application, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Victoria, Vol. 3 (1998) pp. 1326–1332

36.44 H. Kazerooni: Direct-drive active compliant end effector (active RCC), IEEE J. Robotics Autom. **4**(3), 324–333 (1988)

36.45 R.H. Sturges, S. Laowattana: Fine motion planning through constraint network analysis, Proc. Int. Symp. Assem. Task Plan. (1995) pp. 160–170

36.46 D.E. Whitney: Historic perspective and state of the art in robot force control, Int. J. Robotics Res. **6**(1), 3–14 (1987)

36.47 M. Peshkin: Programmed compliance for error corrective assembly, IEEE Trans. Robotics Autom. **6**(4), 473–482 (1990)

36.48 J.M. Schimmels, M.A. Peshkin: Admittance matrix design for force-guided assembly, IEEE Trans. Robotics Autom. **8**(2), 213–227 (1992)

36.49 J.M. Schimmels: A linear space of admittance control laws that guarantees force assembly with friction, IEEE Trans. Robotics Autom. **13**(5), 656–667 (1997)

36.50 S. Hirai, T. Inatsugi, K. Iwata: Learning of admittance matrix elements for manipulative operations, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 763–768

36.51 S. Lee, H. Asada: A perturbation/correlation method for force guided robot assembly, IEEE Trans. Robotics Autom. **15**(4), 764–773 (1999)

36.52 H. Asada: Representation and learning of nonlinear compliance using neural nets, IEEE Trans. Robotics Autom. **9**(6), 863–867 (1993)

36.53 J. Simons, H. Van Brussel, J. De Schutter, J. Verhaert: A self-learning automaton with variable resolution for high precision assembly by industrial robots, IEEE Trans. Autom. Control **27**(5), 1109–1113 (1982)

36.54 V. Gullapalli, J.A. Franklin, H. Benbrahim: Acquiring robot skills via reinforcement learning, IEEE Control Syst. **14**(1), 13–24 (1994)

36.55 Q. Wang, J. De Schutter, W. Witvrouw, S. Graves: Derivation of compliant motion programs based on human demonstration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 2616–2621

Avidbots Ex1016
Page 957 of 2154

36.56 R. Cortesao, R. Koeppe, U. Nunes, G. Hirzinger: Data fusion for robotic assembly tasks based on human skills, IEEE Trans. Robotics Autom. **20**(6), 941–952 (2004)

36.57 K. Ikeuchi, T. Suehiro: Toward an assembly plan from observation. Part I: Task recognition with polyhedral objects, IEEE Trans. Robotics Autom. **10**(3), 368–385 (1994)

36.58 H. Onda, H. Hirokawa, F. Tomita, T. Suehiro, K. Takase: Assembly motion teaching system using position/forcesimulator-generating control program, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 938–945

36.59 H. Onda, T. Suehiro, K. Kitagaki: Teaching by demonstration of assembly motion in VR – nondeterministic search-type motion in the teaching stage, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 3066–3072

36.60 Y. Kobari, T. Nammoto, J. Kinugawa, K. Kosuge: Vision based compliant motion control for part assembly, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 293–298

36.61 T. Lozano-Pérez, M.T. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robot, Int. J. Robotics Res. **31**(1), 3–24 (1984)

36.62 B.R. Donald: *Error Detection and Recovery in Robotics* (Springer, Berlin, Heidelberg 1989)

36.63 J. Canny: On computability of fine motion plans, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 177–182

36.64 M. Erdmann: Using backprojections for fine motion planning with uncertainty, Int. J. Robotics Res. **5**(1), 19–45 (1986)

36.65 J.C. Latombe: *Robot Motion Planning* (Kluwer, Dordrecht 1991)

36.66 B. Dufay, J.C. Latombe: An approach to automatic programming based on inductive learning, Int. J. Robotics Res. **3**(4), 3–20 (1984)

36.67 H. Asada, S. Hirai: Towards a symbolic-level force feedback: Recognition of assembly process states, Proc. Int. Symp. Robotics Res. (1989) pp. 290–295

36.68 J. Xiao, R. Volz: On replanning for assembly tasks using robots in the presence of uncertainties, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 638–645

36.69 J. Xiao: Replanning with compliant rotations in the presence of uncertainties, Proc. Int. Symp. Intell. Control, Glasgow (1992) pp. 102–107

36.70 G. Dakin, R. Popplestone: Simplified fine-motion planning in generalized contact space, Proc. Int. Symp. Intell. Control (1992) pp. 281–287

36.71 G. Dakin, R. Popplestone: Contact space analysis for narrow-clearance assemblies, Proc. Int. Symp. Intell. Control (1993) pp. 542–547

36.72 J. Rosell, L. Basañez, R. Suárez: Compliant-motion planning and execution for robotic assembly, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 2774–2779

36.73 R. Taylor: The Synthesis of Manipulator Control Programs from Task-Level Specifications, Ph.D. Thesis (Stanford University, Stanford 1976)

36.74 R.A. Brooks: Symbolic error analysis and robot planning, Int. J. Robotics Res. **1**(4), 29–68 (1982)

36.75 R.A. Smith, P. Cheeseman: On the representation and estimation of spatial uncertaity, Int. J. Robotics Res. **5**(4), 56–68 (1986)

36.76 S.-F. Su, C. Lee: Manipulation and propagation of uncertainty and verification of applicability of actions in assembly tasks, IEEE Trans. Syst. Man Cybern. **22**(6), 1376–1389 (1992)

36.77 S.-F. Su, C. Lee, W. Hsu: Automatic generation of goal regions for assembly tasks in the presence of uncertainty, IEEE Trans. Robotics Autom. **12**(2), 313–323 (1996)

36.78 J. Hopcroft, G. Wilfong: Motion of objects in contact, Int. J. Robotics Res. **4**(4), 32–46 (1986)

36.79 F. Avnaim, J.D. Boissonnat, B. Faverjon: A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1988) pp. 1656–1661

36.80 R. Brost: Computing metric and topological properties of c-space obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 170–176

36.81 B. Donald: A search algorithm for motion planning with six degrees of freedom, Artif. Intell. **31**(3), 295–353 (1987)

36.82 L. Joskowicz, R.H. Taylor: Interference-free insertion of a solid body into a cavity: An algorithm and a medical application, Int. J. Robotics Res. **15**(3), 211–229 (1996)

36.83 H. Hirukawa: On motion planning of polyhedra in contact, Proc. Workshop Algorithm. Found. Robotics, Toulouse (1996) pp. 381–391

36.84 S.J. Buckley: Planning compliant motion strategies, Proc. Int. Symp. Intell. Control (1988) pp. 338–343

36.85 E. Sacks: Path planning for planar articulated robots using configuration spaces and compliant motion, IEEE Trans. Robotics Autom. **19**(3), 381–390 (2003)

36.86 C. Laugier: Planning fine motion strategies by reasoning in the contact space, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 653–659

36.87 B.J. McCarragher, H. Asada: A discrete event approach to the control of robotic assembly tasks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 331–336

36.88 B.J. McCarragher, H. Asada: The discrete event modeling and trajectory planning of robotic assembly tasks, ASME J. Dyn. Syst. Meas. Control **117**, 394–400 (1995)

36.89 R. Suárez, L. Basañez, J. Rosell: Using configuration and force sensing in assembly task planning and execution, Proc. Int. Symp. Assem. Task Plan. (1995) pp. 273–279

36.90 H. Hirukawa, Y. Papegay, T. Matsui: A motion planning algorithm for convex polyhedra in contact under translation and rotation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Diego (1994) pp. 3020–3027

36.91 X. Ji, J. Xiao: Planning motion compliant to complex contact states, Int. J. Robotics Res. **20**(6), 446–465 (2001)

Avidbots Ex1016
Page 958 of 2154

36.92 L.E. Kavraki, P. Svestka, J.C. Latombe, M. Overmars: Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robotics Autom. **12**(4), 566–580 (1996)

36.93 B. Hannaford, P. Lee: Hidden Markov model analysis of force/torque information in telemanipulation, Int. J. Robotics Res. **10**(5), 528–539 (1991)

36.94 G.E. Hovland, B.J. McCarragher: Hidden Markov models as a process monitor in robotic assembly, Int. J. Robotics Res. **17**(2), 153–168 (1998)

36.95 T. Takahashi, H. Ogata, S. Muto: A method for analyzing human assembly operations for use in automatically generating robot commands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2, Atlanta (1993) pp. 695–700

36.96 P. Sikka, B.J. McCarragher: Rule-based contact monitoring using examples obtained by task demonstration, Proc. 15th Int. Jt. Conf. Artif. Intell., Nagoya (1997) pp. 514–521

36.97 E. Cervera, A. Del Pobil, E. Marta, M. Serna: Perception-based learning for motion in contact in task planning, J. Intell. Robots Syst. **17**(3), 283–308 (1996)

36.98 L.M. Brignone, M. Howarth: A geometrically validated approach to autonomous robotic assembly, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Lausanne (2002) pp. 1626–1631

36.99 M. Nuttin, J. Rosell, R. Suárez, H. Van Brussel, L. Basañez, J. Hao: Learning approaches to contact estimation in assembly tasks with robots, Proc. 3rd Eur. Workshop Learn. Robotics, Heraklion (1995)

36.100 L.J. Everett, R. Ravari, R.A. Volz, M. Skubic: Generalized recognition of single-ended contact formations, IEEE Trans. Robotics Autom. **15**(5), 829–836 (1999)

36.101 M. Skubic, R.A. Volz: Identifying single-ended contact formations from force sensor patterns, IEEE Trans. Robotics Autom. **16**(5), 597–603 (2000)

36.102 S. Hirai, H. Asada: Kinematics and statics of manipulation using the theory of polyhedral convex cones, Int. J. Robotics Res. **12**(5), 434–447 (1993)

36.103 H. Hirukawa, T. Matsui, K. Takase: Automatic determination of possible velocity and applicable force of frictionless objects in contact from a geometric model, IEEE Trans. Robotics Autom. **10**(3), 309–322 (1994)

36.104 B.J. McCarragher, H. Asada: Qualitative template matching using dynamic process models for state transition recognition of robotic assembly, ASME J. Dyn. Syst. Meas. Control **115**(2), 261–269 (1993)

36.105 T.M. Schulteis, P.E. Dupont, P.A. Millman, R.D. Howe: Automatic identification of remote environments, Proc. ASME Dyn. Syst. Control Div., Atlanta (1996) pp. 451–458

36.106 A.O. Farahat, B.S. Graves, J.C. Trinkle: Identifying contact formations in the presence of uncertainty, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Pittsburg (1995) pp. 59–64

36.107 J. Xiao, L. Zhang: Contact constraint analysis and determination of geometrically valid contact formations from possible contact primitives, IEEE Trans. Robotics Autom. **13**(3), 456–466 (1997)

36.108 H. Mosemann, T. Bierwirth, F.M. Wahl, S. Stoeter: Generating polyhedral convex cones from contact graphs for the identification of assembly process states, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 744–749

36.109 J. Xiao, L. Zhang: Towards obtaining all possible contacts – Growing a polyhedron by its location uncertainty, IEEE Trans. Robotics Autom. **12**(4), 553–565 (1996)

36.110 M. Spreng: A probabilistic method to analyze ambiguous contact situations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 543–548

36.111 N. Mimura, Y. Funahashi: Parameter identification of contact conditions by active force sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Diego (1994) pp. 2645–2650

36.112 B. Eberman: A model-based approach to Cartesian manipulation contact sensing, Int. J. Robotics Res. **16**(4), 508–528 (1997)

36.113 T. Debus, P. Dupont, R. Howe: Contact state estimation using multiple model estimation and Hidden Markov model, Int. J. Robotics Res. **23**(4-5), 399–413 (2004)

36.114 J. De Geeter, H. Van Brussel, J. De Schutter, M. Decréton: Recognizing and locating objects with local sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 3478–3483

36.115 J. De Schutter, H. Bruyninckx, S. Dutré, J. De Geeter, J. Katupitiya, S. Demey, T. Lefebvre: Estimating first-order geometric parameters and monitoring contact transitions during force-controlled compliant motions, Int. J. Robotics Res. **18**(12), 1161–1184 (1999)

36.116 T. Lefebvre, H. Bruyninckx, J. De Schutter: Polyhedral contact formation identification for autonomous compliant motion: Exact nonlinear bayesian filtering, IEEE Trans. Robotics **21**(1), 124–129 (2005)

36.117 T. Lefebvre, H. Bruyninckx, J. De Schutter: Online statistical model recognition and state estimation for autonomous compliant motion systems, IEEE Trans. Syst. Man Cybern. C **35**(1), 16–29 (2005)

36.118 K. Gadeyne, T. Lefebvre, H. Bruyninckx: Bayesian hybrid model-state estimation applied to simultaneous contact formation recognition and geometrical parameter estimation, Int. J. Robotics Res. **24**(8), 615–630 (2005)

36.119 K. Kitagaki, T. Ogasawara, T. Suehiro: Methods to detect contact state by force sensing in an edge mating task, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 701–706

36.120 T. Lefebvre, H. Bruyninckx, J. De Schutter: Task planning with active sensing for autonomous compliant motion, Int. J. Robotics Res. **24**(1), 61–82 (2005)

36.121 T. Debus, P. Dupont, R. Howe: Distinguishability and identifiability testing of contact state models, Adv. Robotics **19**(5), 545–566 (2005)

Avidbots Ex1016
Page 959 of 2154

36.122 W. Meeussen, J. Xiao, J. De Schutter, H. Bruyninckx, E. Staffetti: Automatic verification of contact states taking into account manipulator constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 3583–3588

36.123 N.S. Bedrossian: Classification of singular configurations for redundant manipulators, Proc. 1990 IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 818–823

36.124 F.-T. Cheng, T.-L. Hour, Y.-Y. Sun, T.-H. Chen: Study and resolution of singularities for a 6-DOF PUMA manipulator, IEEE Trans. Syst. Man Cybern. B **27**(2), 332–343 (1997)

36.125 A. Sarić, J. Xiao, J. Shi: Robotic surface assembly via contact state transitions, Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE) (2013) pp. 954–959

36.126 J. Park, R. Cortesao, O. Khatib: Multi-contact compliant motion control for robotic manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 4789–4794

36.127 W. Meeussen, E. Staffetti, H. Bruyninckx, J. Xiao, J. De Schutter: Integration of planning and execution in force controlled compliant motion, J. Robotics Auton. Syst. **56**(5), 437–450 (2008)

36.128 A. Stolt, M. Linderoth, A. Robertsson, R. Johansson: Force controlled robotic assembly without a force sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (2012) pp. 1538–1543

36.129 S. Tachi, T. Sakaki, H. Arai, S. Nishizawa, J.F. Pelaez-Polo: Impedance control of a direct-drive manipulator without using force sensors, Adv. Robotics **5**(2), 183–205 (1990)

36.130 M. Van Damme, B. Beyl, V. Vanderborght, V. Grosu, R. Van Ham, I. Vanderniepen, A. Matthys, D. Lefeber: Estimating robot end-effector force from noisy actuator torque measurements, Proc. Int. Conf. Robotics Autom. (ICRA), Shanghai (2011) pp. 1108–1113

36.131 D.E. Koditschek: Exact robot navigation by means of potential functions: Some topological considerations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1–6

36.132 E. Rimon, D.E. Koditschek: Exact robot navigation using artificial potential fields, IEEE Trans. Robotics Autom. **8**(5), 501–518 (1992)

36.133 E. Rimon, D.E. Koditschek: The construction of analytic diffeomorphisms for exact robot navigation on star worlds, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 21–26

36.134 J. Barraquand, J.-C. Latombe: Robot motion planning: A distributed representation approach, Int. J. Robotics Res. **10**(6), 628–649 (1991)

36.135 C.I. Connolly, J.B. Burns, R. Weiss: Path planning using Laplace's equation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 2102–2106

36.136 C.I. Connolly, R.A. Grupen: One the applications of harmonic functions to robotics, J. Robotics Syst. **10**(7), 931–946 (1993)

36.137 S.H.J. Feder, E.J.-J. Slotine: Real-time path planning using harmonic potentials in dynamic environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Albuquerque (1997) pp. 811–874

36.138 J.-O. Kim, P. Khosla: Real-time obstacle avoidance using harmonic potential functions, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Sacramento (1991) pp. 790–796

36.139 K. Sato: Collision avoidance in multi-dimensional space using Laplace potential, Proc. 15th Conf. Robotics Soc. Jpn. (1987) pp. 155–156

36.140 S.M. LaValle, P. Konkimalla: Algorithms for computing numerical optimal feedback motion strategies, Int. J. Robotics Res. **20**(9), 729–752 (2001)

36.141 A. van der Schaft, H. Schumacher: *An Introduction to Hybrid Dynamical Systems* (Springer, Belin, Heidelberg 2000)

36.142 W. Choi, J.-C. Latombe: A reactive architecture for planning and executing robot motions with incomplete knowledge, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 1, Osaka (1991) pp. 24–29

36.143 D.C. Conner, H. Choset, A.A. Rizzi: Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies, Proc. Robotics Sci. Syst., Philadelphia (2006)

36.144 D.C. Conner, A.A. Rizzi, H. Choset: Composition of local potential functions for global robot control and navigation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas (2003) pp. 3546–3551

36.145 S.R. Lindemann, S.M. LaValle: Smooth feedback for car-like vehicles in polygonal environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Rome (2007)

36.146 L. Yang, S.M. LaValle: The sampling-based neighborhood graph: A framework for planning and executing feedback motion strategies, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Taipei (2003)

36.147 C. Belta, V. Isler, G.J. Pappas: Discrete abstractions for robot motion planning and control in polygonal environments, IEEE Trans. Robotics Autom. **21**(5), 864–871 (2005)

36.148 R.R. Burridge, A.A. Rizzi, D.E. Koditschek: Sequential composition of dynamically dexterous robot behaviors, Int. J. Robotics Res. **18**(6), 534–555 (1999)

36.149 S.R. Lindemann, S.M. LaValle: Computing smooth feedback plans over cylindrical algebraic decompositions, Proc. Robotics Sci. Syst. (RSS), Philadephia (2006)

36.150 Y. Yang, O. Brock: Elastic roadmaps: Globally task-consitent motion for autonomous mobile manipulation, Proc. Robotics Sci. Syst. (RSS), Philadephia (2006)

36.151 S. Quinlan, O. Khatib: Elastic bands: Connecting path planning and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2, Atlanta (1993) pp. 802–807

36.152 M. Khatib, H. Jaouni, R. Chatila, J.-P. Laumond: How to implement dynamic paths, Proc.

Avidbots Ex1016
Page 960 of 2154

Int. Symp. Exp. Robotics (1997) pp. 225–236, Preprints
36.153 O. Brock, O. Khatib: Elastic strips: A framework for motion generation in human environments, Int. J. Robotics Res. **21**(12), 1031–1052 (2002)
36.154 M.H. Raibert, J.J. Craig: Hybrid position/force control of manipulators, J. Dyn. Syst. Meas. Control **103**(2), 126–133 (1981)
36.155 R. Featherstone, S. Sonck, O. Khatib: A general contact model for dynamically-decoupled force/motion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 3281–3286
36.156 J. Park, O. Khatib: Multi-link multi-contact force control for manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005)
36.157 D.C. Ruspini, O. Khatib: A framework for multi-contact multi-body dynamic simulation and haptic display, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Takamatsu (2000) pp. 1322–1327
36.158 K.-S. Chang, R. Holmberg, O. Khatib: The augmented object model: Cooperative manipluation and parallel mechanism dynamics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 470–475
36.159 O. Khatib: Object Manipulation in a multi-effector robot system. In: *Robotics Research 4*, ed. by R. Bolles, B. Roth (MIT Press, Cambridge 1988) pp. 137–144
36.160 D. Williams, O. Khatib: The virtual linkage: A model for internal forces in multi-grasp manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1, Altanta (1993) pp. 1030–1035
36.161 J.A. Adams, R. Bajcsy, J. Kosecka, V. Kuma, R. Mandelbaum, M. Mintz, R. Paul, C. Wang, Y. Yamamoto, X. Yun: Cooperative material handling by human and robotic agents: Module development and system synthesis, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Pittsburgh (1995) pp. 200–205
36.162 S. Hayati: Hybrid postiion/force control of multi-arm cooperating robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 82–89
36.163 D. Jung, G. Cheng, A. Zelinsky: Experiments in realizing cooperation between autonomous mobile robots, Proc. Int. Symp. Exp. Robotics (1997) pp. 513–524
36.164 T.-J. Tarn, A.K. Bejczy, X. Yun: Design of dynamic control of two cooperating robot arms: Closed chain formulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 7–13
36.165 M. Uchiyama, P. Dauchez: A symmetric hybrid position/force control scheme for the coordination of two robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Philadelphia (1988) pp. 350–356
36.166 X. Yun, V.R. Kumar: An approach to simultaneious control fo trajecotry and integraction forces in dual-arm configurations, IEEE Trans. Robotics Autom. **7**(5), 618–625 (1991)
36.167 Y.F. Zheng, J.Y.S. Luh: Joint torques for control of two coordinated moving robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 1375–1380
36.168 T. Bretl, Z. McCarthy: Quasi-static manipulation of a Kirchhoff elastic rod based on a geometric analysis of equilibrium configurations, Int. J. Robotics Res. **33**(1), 48–68 (2014)
36.169 J. Russakow, O. Khatib, S.M. Rock: Extended operational space formation for serial-to-parallel chain (branching) manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1, Nagoya (1995) pp. 1056–1061
36.170 K.-S. Chang, O. Khatib: Operational space dynamics: Efficient algorithms for modelling and control of branching mechanisms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 850–856
36.171 K. Kreutz-Delgado, A. Jain, G. Rodriguez: Recursive formulation of operational space control, Int. J. Robotics Res. **11**(4), 320–328 (1992)
36.172 M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, R. Siegwart: Quadrupedal locomotion using hierarchical operational space control, Int. J. Robotics Res. **33**(8), 1047–1062 (2014)
36.173 B. Bayle, J.-Y. Fourquet, M. Renaud: A coordination strategy for mobile manipulation, Proc. Int. Conf. Intell. Auton. Syst., Venice (2000) pp. 981–988
36.174 B. Bayle, J.-Y. Fourquet, M. Renaud: Generalized path generation for a mobile manipulator, Proc. Int. Conf. Mech. Des. Prod., Cairo (2000) pp. 57–66
36.175 B. Bayle, J.-Y. Fourquet, M. Renaud: Using manipulability with nonholonomic mobile manipulators, Proc. Int. Conf. Field Serv. Robotics, Helsinki (2001) pp. 343–348
36.176 J. Peters, S. Schaal: Reinforcement learning for operational space control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Rome (2007)
36.177 T. Simeon, J. Cortes, A. Sahbani, J.P. Laumond: A manipulation planner for pick and place operations under continuous grasps and placements, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)
36.178 A. Miller, P. Allen: Examples of 3D grasp quality computations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (1999)
36.179 A.T. Miller: Graspit: A Versatile Simulator for Robotic Grasping, Ph.D. Thesis (New York, Columbia University 2001)
36.180 G. Wilfong: Motion panning in the presence of movable obstacles, Proc. ACM Symp. Comput. Geom. (1988) pp. 279–288
36.181 M. Erdmann, T. Lozano-Perez: On multiple moving objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 1419–1424
36.182 P.C. Chen, Y.K. Hwang: Pracitcal path planning among movable obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1991) pp. 444–449
36.183 M. Stilman, J.J. Kuffner: Navigation among movable obstacles: Real-time reasoning in complex environments, Int. J. Hum. Robotics **2**(4), 1–24 (2005)
36.184 M. Stilman, J.-U. Shamburek, J.J. Kuffner, T. Asfour: Manipulation planning among movable ob-

Avidbots Ex1016
Page 961 of 2154

stacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)
36.185 D. Nieuwenhuisen, A.F. van der Stappen, M.H. Overmars: An effective framework for path planning amidst movable obstacles, Proc. Workshop Algorithm. Found. Robotics (2006)
36.186 K.M. Lynch, M.T. Mason: Stable pushing: Mechanics, controllability, and planning, Int. J. Robotics Res. **15**(6), 533–556 (1996)
36.187 P. Tang, J. Xiao: Generation of point-contact state space between strictly curved objects. In: *Robotics Science and Systems II*, ed. by G.S. Sukhatme, S. Schaal, W. Burgard, D. Fox (MIT Press, Cambridge 2007) pp. 239–246
36.188 H. Nakagaki, K. Kitagaki, T. Ogasawara, H. Tsukune: Study of deformation and insertion tasks of a flexible wire, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 2397–2402
36.189 W. Kraus Jr., B.J. McCarragher: Case studies in the manipulation of flexible parts using a hybrid position/force approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 367–372
36.190 J.Y. Kim, D.J. Kang, H.S. Cho: A flexible parts assembly algorithm based on a visual sensing system, Proc. Int. Symp. Assem. Task Plan. (2001) pp. 417–422
36.191 B.J. Unger, A. Nocolaidis, P.J. Berkelman, A. Thompson, R.L. Klatzky, R.L. Hollis: Comparison of 3-D haptic peg-in-hole tasks in real and virtual environments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1751–1756

Avidbots Ex1016
Page 962 of 2154

Multimedia Contents

37. Contact Modeling and Manipulation

Imin Kao, Kevin M. Lynch, Joel W. Burdick

Robotic manipulators use contact forces to grasp and manipulate objects in their environments. Fixtures rely on contacts to immobilize workpieces. Mobile robots and humanoids use wheels or feet to generate the contact forces that allow them to locomote. Modeling of the *contact interface*, therefore, is fundamental to analysis, design, planning, and control of many robotic tasks.

This chapter presents an overview of the modeling of contact interfaces, with a particular focus on their use in manipulation tasks, including graspless or *nonprehensile* manipulation modes such as pushing. Analysis and design of grasps and fixtures also depends on contact modeling, and these are discussed in more detail in Chap. 38. Sections 37.2–37.5 focus on rigid-body models of contact. Section 37.2 describes the kinematic constraints caused by contact, and Sect. 37.3 describes the contact forces that may arise with Coulomb friction. Section 37.4 provides examples of analysis of multicontact manipulation tasks with rigid bodies and Coulomb friction. Section 37.5 extends the analysis to manipulation by pushing. Section 37.6 introduces modeling of contact interfaces, kinematic duality, and pressure distribution and soft contact interface. Section 37.7 describes the concept of the friction limit surface and illustrates it with an example demonstrating the construction of a limit surface for a soft contact. Finally, Sect. 37.8 discusses how these more accurate models can be used in fixture analysis and design.

37.1 **Overview** 931
37.1.1 Choosing a Contact Model 932
37.1.2 Grasp/Manipulation Analysis 932
37.2 **Kinematics of Rigid-Body Contact** 932
37.2.1 Contact Constraints 933
37.2.2 Collections of Parts 934
37.2.3 Graphical Planar Methods 935
37.3 **Forces and Friction** 936
37.3.1 Graphical Planar Methods 937
37.3.2 Duality of Contact Wrenches and Twist Freedoms 937
37.4 **Rigid-Body Mechanics with Friction** 939
37.4.1 Complementarity 939
37.4.2 Quasistatic Assumption 940
37.4.3 Examples 940
37.5 **Pushing Manipulation** 942
37.6 **Contact Interfaces and Modeling** 943
37.6.1 Modeling of Contact Interface 943
37.6.2 Pressure Distribution at Contacts 945
37.7 **Friction Limit Surface** 946
37.7.1 The Friction Limit Surface at a Soft Contact Interface 947
37.7.2 Example of Constructing a Friction Limit Surface 948
37.8 **Contacts in Grasping and Fixture Designs** 949
37.8.1 Contact Stiffness of Soft Fingers 949
37.8.2 Application of Soft Contact Theory to Fixture Design 950
37.9 **Conclusions and Further Reading** 950
Video-References 951
References 951

37.1 Overview

A contact model characterizes both the forces that can be transmitted through the contact as well as the allowed relative motions of the contacting bodies. These characteristics are determined by the geometry of the con-

Avidbots Ex1016
Page 963 of 2154

tacting surfaces and the material properties of the parts, which dictate friction and possible contact deformation.

37.1.1 Choosing a Contact Model

The choice of a contact model largely depends upon the application or analysis that must be carried out. When appropriate analytical models are used, one can determine if a manipulation plan or fixture design meets desired functional requirements within the model's limits.

Rigid-Body Models

Many approaches to manipulation, grasp, and fixture analysis are based on rigid-body models. In the rigid-body model, no deformations are allowed at the points or surfaces of contact between two bodies. Instead, contact forces arise from two sources: the constraint of incompressibility and impenetrability between the rigid bodies, and surface frictional forces. Rigid-body models are straightforward to use, lead to computationally efficient planning algorithms, and are compatible with solid-modeling software systems. Rigid-body models are often appropriate for answering qualitative questions such as *will this fixture be able to hold my workpiece*? and for problems involving stiff parts with low to moderate contact forces.

Rigid-body models are not capable of describing the full range of contact phenomena, however; for example, rigid-body models cannot predict the individual contact forces of a multiple-contact fixture (the static indeterminacy problem [37.1, 2]). Furthermore, workpieces held in fixtures experience non-negligible deformations in many high-force manufacturing operations [37.3–6]. These deformations, which crucially impact machining accuracy, cannot be determined from rigid-body models. Also, a rigid-body model augmented with a Coulomb friction model can lead to mechanics problems that have no solution or multiple solutions [37.7–15]. To overcome the limitations inherent in the rigid-body model, one must introduce compliance into the contact model.

Compliant Models

A *compliant* contact deforms under the influence of applied forces. The forces of interaction at the contact are derived from the *compliance* or *stiffness* model. While compliant contact models are typically more complicated, they have several advantages: they overcome the static indeterminacy inherent to rigid-body models and they predict the deformations of grasped or fixtured parts during loading.

A detailed model of the deformations of real materials can be quite complex. Consequently, for analysis we often introduce *lumped-parameter* or *reduced-order* compliance models having a limited number of variables. In this chapter we summarize a reduced-order *quasi-rigid-body* approach to modeling that can model a variety of compliant materials in a way that is consistent with both the solid mechanics literature and conventional robot analysis and planning paradigms.

Finally, three-dimensional finite-element models [37.16–18] or similar ideas [37.19, 20] can be used to analyze workpiece deformations and stresses in fixtures. While accurate, these numerical approaches have some drawbacks, for example, the grasp stiffness matrix can only be found through difficult numerical procedures. Stiffness matrices are often needed to compute quality measures that are the basis for optimal grasping plans or fixturing designs [37.21, 22]. Thus, these numerical approaches are better suited for verifying final fixture designs.

37.1.2 Grasp/Manipulation Analysis

Once a contact model has been chosen, we can use it to analyze tasks involving multiple contacts. If a part is subject to multiple contacts, the kinematic constraints and force freedoms due to the individual contacts must be combined. This combined analysis facilitates manipulation planning – choosing the contact locations, and possibly the motions or forces applied by those contacts, to achieve the desired behavior of the part. A prime example is the grasping or fixturing problem: choosing contact locations, and possibly contact forces, to prevent motion of a part in the face of external disturbances. This well-studied topic is discussed in greater detail in Chap. 38. Other examples include problems of partial constraint, such as pushing a part or inserting a peg into a hole.

37.2 Kinematics of Rigid-Body Contact

Contact kinematics is the study of how two or more parts can move relative to each other while respecting the rigid-body impenetrability constraint. It also classifies motion in contact as either rolling or slipping.

Consider two rigid bodies whose position and orientation (configuration) is given by the local coordinate column vectors $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$, respectively. Writing the composite configuration as $\boldsymbol{q} = (\boldsymbol{q}_1^{\mathrm{T}}, \boldsymbol{q}_2^{\mathrm{T}})^{\mathrm{T}}$, we de-

fine a *distance* function $d(\boldsymbol{q})$ between the parts that is positive when they are separated, zero when they are touching, and negative when they are in penetration. When $d(\boldsymbol{q}) > 0$, there are no constraints on the motions of the parts. When the parts are in contact ($d(\boldsymbol{q}) = 0$), we look at the time derivatives $\dot{d}$, $\ddot{d}$, etc., to determine if the parts stay in contact or break apart as they follow a trajectory $\boldsymbol{q}(t)$. This can be determined by the following table of possibilities

d	$\dot{d}$	$\ddot{d}$	...
> 0			no contact,
< 0			infeasible (penetration),
$= 0$	> 0		breaking contact,
$= 0$	< 0		infeasible (penetration),
$= 0$	$= 0$	> 0	breaking contact,
$= 0$	$= 0$	< 0	infeasible (penetration),
etc.			

The contact is maintained only if all time derivatives are zero.

The first two time derivatives are written

$$\dot{d} = \left(\frac{\partial d}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \dot{\boldsymbol{q}} \,, \tag{37.1}$$

$$\ddot{d} = \dot{\boldsymbol{q}}^{\mathrm{T}} \frac{\partial^2 d}{\partial \boldsymbol{q}^2} \dot{\boldsymbol{q}} + \left(\frac{\partial d}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \ddot{\boldsymbol{q}} \,. \tag{37.2}$$

The terms $\partial d/\partial \boldsymbol{q}$ and $\partial^2 d/\partial \boldsymbol{q}^2$ carry information about the local contact geometry. The former corresponds to the contact normal, while the latter corresponds to the relative curvature of the parts at the contact.

If contact is maintained, we can classify the contact as slipping or rolling. Analogous to the table above, the contact is rolling if and only if there is zero relative tangential velocity, acceleration, etc., between the contact points on the parts. If the relative tangential velocity is nonzero, the parts are slipping; if the relative velocity is zero but relative tangential acceleration or (higher-order derivatives) is not, slipping is incipient.

In this section we focus on a *first-order* analysis of contact kinematics. A first-order analysis concludes that contact is maintained if $d(\boldsymbol{q}) = 0$ and $\dot{d} = 0$. This local linearization of contact kinematics focuses on the velocity $\dot{\boldsymbol{q}}$ and the contact normal in $\partial d/\partial \boldsymbol{q}$; higher-order spatial derivatives of the contact geometry (curvature, etc.) are not considered. While this is a good starting point, it may occasionally lead to erroneous conclusions. For example, *Rimon* and *Burdick* [37.23–25] showed that a first-order analysis may incorrectly predict mobility of a part in a fixture when a second-order analysis shows that it is in fact completely constrained.

Analysis of the kinematics of rolling contact of parameterized surfaces can be found in [37.26]; see also [37.27–30].

37.2.1 Contact Constraints

As described in Chap. 2, a rigid body in space has six degrees of freedom, specified by the location of the origin of a coordinate frame P affixed to the part and the orientation of this coordinate frame relative to an inertial frame O fixed in the world. Let ${}^{O}\boldsymbol{p}_P \in \mathbb{R}^3$ be the position of the part center of mass and ${}^{O}\mathbf{R}_P \in SO(3)$ be the rotation matrix describing the orientation of the part relative to O. The spatial velocity of the part can be written as $\boldsymbol{t} \in \mathbb{R}^6$, sometimes called a *twist*,

$$\boldsymbol{t} = \left(\boldsymbol{\omega}^{\mathrm{T}},\ \boldsymbol{v}^{\mathrm{T}}\right)^{\mathrm{T}} \,,$$

where $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^{\mathrm{T}}$ and $\boldsymbol{v} = (v_x, v_y, v_z)^{\mathrm{T}}$ give the angular velocity and linear velocity of P in the world frame O, respectively, such that $\boldsymbol{\omega}$ satisfies

$${}^{O}\dot{\mathbf{R}}_P = \boldsymbol{\omega} \times {}^{O}\mathbf{R}_P$$

and $\boldsymbol{v}$ satisfies

$$\boldsymbol{v} = {}^{O}\dot{\boldsymbol{p}}_P - \boldsymbol{\omega} \times {}^{O}\boldsymbol{p}_P \,.$$

It is worth taking a moment to really understand the spatial velocity of a body. It consists of the body's angular velocity expressed in the world frame O, along with the linear velocity of a point as if it were rigidly attached to the body but currently at the origin of the world frame. This point need not be physically on the body. In other words, $\boldsymbol{v}$ is not simply ${}^{O}\dot{\boldsymbol{p}}_P$. This notation will simplify the following expressions, where all velocities and forces will be expressed in the common world frame O. (Be aware that twists are sometimes defined in a body frame instead.)

A point contact acting on the part provides a unilateral constraint which prevents the part from locally moving against the contact normal. Let $\boldsymbol{x}$ be the location of the contact in O. The linear velocity of the point on the part in contact is

$$\boldsymbol{v}_{\mathrm{C}} = \boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{x} \,.$$

(We drop the pre-superscripts O for simplicity; for example, ${}^{O}\boldsymbol{x}$ is written simply as $\boldsymbol{x}$.) Let $\hat{\boldsymbol{u}}$ be the unit vector normal pointing into the part (Fig. 37.1). The first-order condition that the part not move into the unilateral constraint can be written

$$\boldsymbol{v}_{\mathrm{C}}^{\mathrm{T}} \hat{\boldsymbol{u}} = (\boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{x})^{\mathrm{T}} \hat{\boldsymbol{u}} \geq 0 \,. \tag{37.3}$$

Avidbots Ex1016
Page 965 of 2154

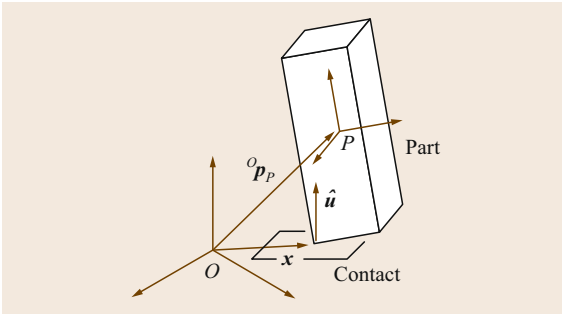


Fig. 37.1 Notation for a part in contact with a manipulator or the environment

In other words, the velocity of the part at C cannot have a component in the opposite direction of the contact normal. To write this another way, define a generalized force or *wrench* $\boldsymbol{w}$ consisting of the torque $\boldsymbol{m}$ and force $\boldsymbol{f}$ acting on the part for a unit force along the contact normal

$$\boldsymbol{w} = \left(\boldsymbol{m}^{\mathrm{T}}, \boldsymbol{f}^{\mathrm{T}}\right)^{\mathrm{T}} = \left[(\boldsymbol{x} \times \hat{\boldsymbol{u}})^{\mathrm{T}}, \hat{\boldsymbol{u}}^{\mathrm{T}}\right]^{\mathrm{T}} .$$

Then (37.3) can be rewritten as

$$\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} \geq 0 . \tag{37.4}$$

If the external constraint point is moving with linear velocity $\boldsymbol{v}_{\mathrm{ext}}$, (37.4) changes to

$$\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} \geq \boldsymbol{v}_{\mathrm{ext}}^{\mathrm{T}} \hat{\boldsymbol{u}} , \tag{37.5}$$

which reduces to (37.4) if the constraint is stationary.

Each inequality of the form (37.5) constrains the velocity of the part to a half-space of its six-dimensional velocity space bounded by the hyperplane $\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} = \boldsymbol{v}_{\mathrm{ext}}^{\mathrm{T}} \hat{\boldsymbol{u}}$. Unioning the set of all constraints, we get a convex polyhedral set of feasible part velocities. A constraint is redundant if its half-space constraint does not change the feasible velocity polyhedron. For a given twist $\boldsymbol{t}$, a constraint is active if

$$\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} = \boldsymbol{v}_{\mathrm{ext}}^{\mathrm{T}} \hat{\boldsymbol{u}} ; \tag{37.6}$$

otherwise the part is breaking contact at that point. In general, the feasible velocity polyhedron for a part can consist of a six-dimensional interior (where no contact constraint is active), five-dimensional hyperfaces, four-dimensional hyperfaces, and so on, down to one-dimensional edges and zero-dimensional points. A part velocity on an n-dimensional facet of the velocity polyhedron indicates that $6-n$ independent (nonredundant) constraints are active.

If all of the constraints are stationary ($\boldsymbol{v}_{\mathrm{ext}} = 0$), then the boundary of each half-space defined by (37.5) passes through the origin of the velocity space, and the feasible velocity set becomes a cone rooted at the origin. Let $\boldsymbol{w}_i$ be the constraint wrench of stationary contact i. Then the feasible velocity cone is

$$V = \{\boldsymbol{t} \,|\, \boldsymbol{t}^{\mathrm{T}} \boldsymbol{w}_i \geq 0 \quad \forall i\} .$$

If the $\boldsymbol{w}_i$ span the six-dimensional generalized force space, or, equivalently, the convex hull of the $\boldsymbol{w}_i$ contains the origin in the interior, then V is the null set, the stationary contacts completely constrain the motion of the part, and we have form closure, as discussed in Chap. 38.

In the discussion above, each constraint (37.5) divides the part velocity space into three categories: a hyperplane of velocities that maintain contact, a half-space of velocities that separate the parts, and a half-space of velocities that cause the parts to penetrate. Velocities where the contact is maintained can be further broken down into two categories: velocities where the part slips over the contact constraint, and velocities where the part sticks or rolls on the constraint. In the latter case, the part velocity satisfies the three equations

$$\boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{x} = \boldsymbol{v}_{\mathrm{ext}} . \tag{37.7}$$

Now we can give each point contact i a label m_i corresponding to the type of contact, called the contact label: `b` if the contact is breaking, `f` if the contact is fixed (including rolling), and `s` if the contact is slipping, i. e., (37.6) is satisfied but (37.7) is not. The *contact mode* for the entire system can be written as the concatenation of the contact labels at the k contacts, $m_1 m_2 \ldots m_k$.

37.2.2 Collections of Parts

The discussion above can be generalized to find the feasible velocities of multiple parts in contact. If parts i and j make contact at a point $\boldsymbol{x}$, where $\hat{\boldsymbol{u}}_i$ points into part i and $\boldsymbol{w}_i = [(\boldsymbol{x} \times \hat{\boldsymbol{u}}_i)^{\mathrm{T}}, \hat{\boldsymbol{u}}_i^{\mathrm{T}}]^{\mathrm{T}}$, then their spatial velocities $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ must satisfy the constraint

$$(\boldsymbol{t}_i - \boldsymbol{t}_j)^{\mathrm{T}} \boldsymbol{w}_i \geq 0 \tag{37.8}$$

to avoid penetration. This is a homogeneous half-space constraint in the composite $(\boldsymbol{t}_i, \boldsymbol{t}_j)$ velocity space. In an assembly of multiple parts, each pairwise contact contributes another constraint in the composite part velocity space, and the result is a polyhedral convex cone of kinematically feasible velocities rooted at the origin of the composite velocity space. The contact mode for the entire assembly is the concatenation of the contact labels at each contact in the assembly.

If there are moving contacts whose motion is prescribed, e.g., robot fingers, the constraints on the motion of the remaining parts will no longer be homogeneous. As a result, the convex polyhedral feasible velocity space is no longer a cone rooted at the origin.

37.2.3 Graphical Planar Methods

When a part is confined to move in the x–y-plane, the twist $\boldsymbol{t}$ reduces to $\boldsymbol{t} = (\omega_x, \omega_y, \omega_z, v_x, v_y, v_z)^{\mathrm{T}} = (0, 0, \omega_z, v_x, v_y, 0)^{\mathrm{T}}$. The point $(-v_y/\omega_z, v_x/\omega_z)$ is called the center of rotation (COR) in the projective plane, and we can represent any planar twist by its COR and rotational velocity ω_z. (Note that the case $\omega_z = 0$ must be treated with care, as it corresponds to a COR at infinity.) This is sometimes useful for graphical purposes: for a single part constrained by stationary fixtures, at least, we can easily draw the feasible twist cone as CORs [37.14, 31].

As an example, Fig. 37.2a shows a planar part standing on a table and being contacted by a robot finger.

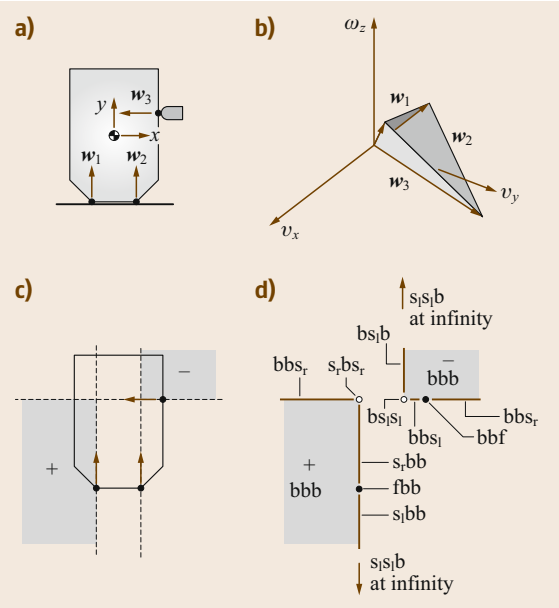


Fig. 37.2 **(a)** A part resting on a table and the three contact constraints. **(b)** The twist cone satisfying the contact constraints. The contact normals are shown with their associated constraint planes. **(c)** The equivalent representation as CORs, shown in *grey*. Note that the lines that extend off to the *left* and to the *bottom wrap around* at infinity and come back in from the *right* and the *top*, respectively, so this COR region should be interpreted as a single connected convex region, equivalent to the twist cone representation in **(b)**. **(d)** The contact modes assigned to each feasible motion. The zero velocity contact mode is fff

This finger is currently stationary, but we will later set it in motion. The finger defines one constraint on the part's motion and the table defines two more. (Note that contact points in the interior of the edge between the part and the table provide redundant kinematic constraints.) The constraint wrenches can be written

$$\boldsymbol{w}_1 = (0, 0, -1, 0, 1, 0)^{\mathrm{T}} ,$$
$$\boldsymbol{w}_2 = (0, 0, 1, 0, 1, 0)^{\mathrm{T}} ,$$
$$\boldsymbol{w}_3 = (0, 0, 1, -1, 0, 0)^{\mathrm{T}} .$$

For a stationary finger, the kinematic constraints yield the feasible twist cone shown in Fig. 37.2b. This region can also be easily visualized in the plane by using the following method: at each contact, draw the contact normal line. Label all points on the normal $\pm$, points to the left of the inward normal $+$, and points to the right $-$. For each contact constraint, all the points labeled $+$ can serve as CORs with positive angular velocity, and all the points labeled $-$ can serve as CORs with negative angular velocity, without violating the contact constraint. After doing this for all the contact normals, keep only the CORs that are consistently labeled. These CORs are a planar representation of the feasible twist cone (Fig. 37.2c).

We can refine this method by assigning contact modes to each feasible COR. For each contact normal, label the COR at the contact point f for fixed, other CORs on the normal line s for slipping, and all other CORs b for breaking contact. The concatenation of these labels gives the part's contact mode for a particular part motion. In the planar case, the label s on the contact normal line can be further refined into $\mathrm{s_r}$ or $\mathrm{s_l}$, indicating whether the part is slipping right or left relative to the constraint. An s COR labeled $+$ above the contact (in the direction of the contact normal) or $-$ below the contact should be relabeled $\mathrm{s_r}$, and an s COR labeled $-$ above the contact or $+$ below should be relabeled $\mathrm{s_l}$ (Fig. 37.2d).

This method can be used readily to determine if the part is in form closure. If there is no COR labeled consistently, then the feasible velocity cone consists of only the zero velocity point, and the part is immobilized by the stationary contacts. This method also makes it clear that at least four contacts are necessary to immobilize the part by the first-order analysis (Chap. 38). This is a weakness of the first-order analysis – curvature effects can be used to immobilize a part with three or even two contacts [37.24]. This weakness can also be seen in Fig. 37.2d. A pure rotation about the COR labeled $\{+, \mathrm{s_r b s_r}\}$ is actually not feasible, but it would be if the part had a small radius of curvature at the contact with the finger. The first-order analysis ignores this curvature.

Avidbots Ex1016
Page 967 of 2154

37.3 Forces and Friction

A commonly used model of friction in robotic manipulation is Coulomb's law [37.32]. This experimental law states that the friction force magnitude f_t in the tangent plane at the contact interface is related to the normal force magnitude f_n by $f_t \le \mu f_n$, where μ is called the friction coefficient. If the contact is sliding, then $f_t = \mu f_n$, and the friction force opposes the direction of motion. The friction force is independent of the speed of sliding.

Often two friction coefficients are defined, a static friction coefficient μ_s and a kinetic (or sliding) friction coefficient μ_k, where $\mu_s \ge \mu_k$. This implies that a larger friction force may be available to resist initial motion, but once motion has begun, the resisting force decreases. Many other friction models have been developed with different functional dependencies on factors such as the speed of sliding and the duration of static contact before sliding. All of these are aggregate models of complex microscopic behavior. For simplicity, we will assume the simplest Coulomb friction model with a single friction coefficient μ. This model is reasonable for hard, dry materials. The friction coefficient depends on the two materials in contact, and typically ranges from 0.1 to 1.

Figure 37.3a shows that this friction law can be interpreted in terms of a *friction cone*. The set of all forces that can be applied to the disk by the supporting line is constrained to be inside this cone. Correspondingly, any force the disk applies to the support is inside the negative of the cone. The half-angle of the cone is $\beta = \tan^{-1}\mu$, as shown in Fig. 37.4. If the disk slips to the left on the support, the force the support applies to it acts on the right edge of the friction cone, with a magnitude determined by the normal force.

If we choose a coordinate frame, the force $\boldsymbol{f}$ applied to the disk by the support can be expressed as a wrench $\boldsymbol{w} = [(\boldsymbol{x} \times \boldsymbol{f})^T, \boldsymbol{f}^T]^T$, where $\boldsymbol{x}$ is the contact location. Thus the friction cone turns into a wrench cone, as shown in Fig. 37.3b. The two edges of the planar friction cone give two half-lines in the wrench space, and the wrenches that can be transmitted to the part through the contact are all nonnegative linear combinations of basis vectors along these edges. If $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ are basis vectors for these wrench cone edges, we write the wrench cone as

$$\mathcal{WC} = \{k_1 \boldsymbol{w}_1 + k_2 \boldsymbol{w}_2 \mid k_1, k_2 \ge 0\} .$$

If there are multiple contacts acting on a part, then the total set of wrenches that can be transmitted to the part through the contacts is the nonnegative linear combination of all the individual wrench cones $\mathcal{WC}_i$,

$$\mathcal{WC} = \text{pos}(\{\mathcal{WC}_i\}) = \left\{ \sum_i k_i \boldsymbol{w}_i | \boldsymbol{w}_i \in \mathcal{WC}_i, k_i \ge 0 \right\} .$$

This composite wrench cone is a convex polyhedral cone rooted at the origin. An example composite

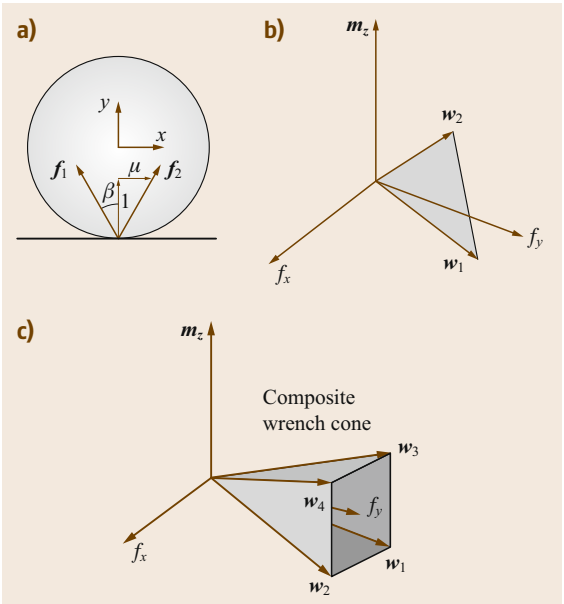


Fig. 37.3 (a) A planar friction cone. (b) The corresponding wrench cone. (c) An example composite wrench cone resulting from two frictional contacts

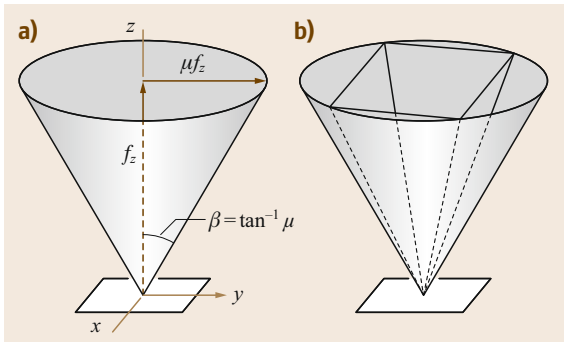


Fig. 37.4 (a) A spatial friction cone. The half-angle of the cone is $\beta = \tan^{-1}\mu$. (b) An inscribed pyramidal approximation to the friction cone. A more accurate inscribed pyramidal approximation can be used by increasing the number of faces of the pyramid. Depending on the application, a circumscribed pyramid could be used instead of an inscribed pyramid

wrench cone arising from two planar frictional contacts is shown in Fig. 37.3c. If the composite wrench cone is the entire wrench space, then the contacts can provide a force-closure grasp (Chap. 38).

In the spatial case, the friction cone is a circular cone, defined by

$$\sqrt{f_x^2 + f_y^2} \leq \mu f_z \,, \quad f_z \geq 0 \,, \tag{37.9}$$

when the contact normal is in the $+z$-direction (Fig. 37.4). The resulting wrench cone, and composite wrench cone $\text{pos}(\{\mathcal{W}C_i\})$ for multiple contacts, is a convex cone rooted at the origin, but it is not polyhedral. For computational purposes, it is common to approximate circular friction cones as pyramidal cones, as shown in Fig. 37.4. Then individual and composite wrench cones become polyhedral convex cones in the six-dimensional wrench space.

If a contact or set of contacts acting on a part is ideally force controlled, the wrench $\boldsymbol{w}_{\text{ext}}$ specified by the controller must lie within the composite wrench cone corresponding to those contacts. Because these force-controlled contacts choose a subset of wrenches (possibly a single wrench) from this wrench cone, the total composite wrench cone that can act on the part (including other non-force-controlled contacts) may no longer be a homogeneous cone rooted at the origin. This is roughly analogous to the case of velocity-controlled contacts in Sect. 37.2.1, which results in a feasible twist set for the part that is not a cone rooted at the origin. Ideal robot manipulators may be controlled by position control, force control, hybrid position–force control, or some other scheme. The control method must be compatible with the parts' contacts with each other and the environment to prevent excessive forces [37.33].

37.3.1 Graphical Planar Methods

Just as homogeneous twist cones for planar problems can be represented as convex signed (+ or −) COR regions in the plane, homogeneous wrench cones for planar problems can be represented as convex signed regions in the plane. This is called *moment labeling* [37.14, 34]. Given a collection of lines of force in the plane (e.g., the edges of friction cones from a set of point contacts), the set of all nonnegative linear combinations of these can be represented by labeling all the points in the plane with either a + if all resultants make nonnegative moment about that point, a − if all make nonpositive moment about that point, a ± if all make zero moment about that point, and a blank label if there exist resultants making positive moment and resultants making negative moment about that point.

The idea is best illustrated by an example. In Fig. 37.5a, a single line of force is represented by labeling the points to the left of the line with a + and points to the right of the line with a −. Points on the line are labeled ±. In Fig. 37.5b, another line of force is added. Only the points in the plane that are consistently labeled for both lines of force retain their labels; inconsistently labeled points lose their labels. Finally, a third line of force is added in Fig. 37.5c. The result is a single region labeled +. A nonnegative combination of the three lines of force can create any line of force in the plane that passes around this region in a counterclockwise sense. This representation is equivalent to a homogeneous convex wrench cone representation.

37.3.2 Duality of Contact Wrenches and Twist Freedoms

Our discussion of kinematic constraints and friction should make it apparent that, for any point contact and contact label, the number of equality constraints on the part's motion caused by that contact is equal to the number of wrench freedoms it provides. For example, a breaking contact `b` provides zero equality constraints on the part motion and also allows no contact force. A fixed contact `f` provides three motion constraints (the motion of a point on the part is specified) and three freedoms on the contact force: any wrench in the interior of the contact wrench cone is consistent with the contact mode. Finally, a slipping contact `s` provides one equality motion constraint (one equation on the part's motion must be satisfied to maintain the contact), and for a given motion satisfying the constraint, the contact wrench has only one freedom, the magnitude of the contact wrench on the edge of the friction cone and opposite the slipping direction. In the planar case, the motion constraints and wrench freedoms for `b`, `s`, and `f` contacts are zero, one, and two, respectively.

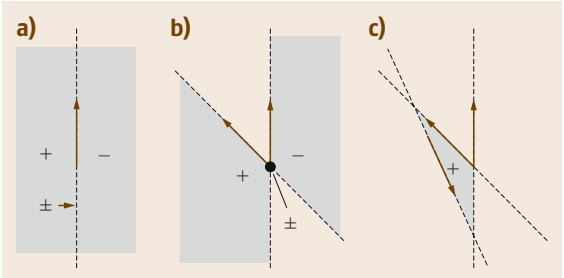


Fig. 37.5 **(a)** Representing a line of force by moment labels. **(b)** Representing the nonnegative linear combinations of two lines of force by moment labels. **(c)** Nonnegative linear combinations of three lines of force

Table 37.1 Duality of the kinematics of contact interface. The force/motion relationships for finger and grasped object are dual to each other in the joint space, contact interface, and the Cartesian space of the object. In this table, the notation of θ, $\boldsymbol{x}_{\mathrm{f}}$, $\boldsymbol{x}_{\mathrm{p}}$, and $\boldsymbol{x}_{\mathrm{b}}$ are the displacement in the joint space, the Cartesian space at the contact point on the finger, the Cartesian space at the contact point on the object, and the reference frame on the object, respectively, with $\boldsymbol{x}_{\mathrm{tr}}$ denoting the transmitted components. The δ prefix denotes infinitesimal changes in the named coordinates. The forces with corresponding subscripts are referenced to the corresponding coordinates as those of the displacement described before

Motion	$\mathbf{J}_\theta \quad \delta\boldsymbol{\theta} = \delta\boldsymbol{x}_{\mathrm{f}}$	$\mathbf{H} \quad \delta\boldsymbol{x}_{\mathrm{f}} = \delta\boldsymbol{x}_{\mathrm{tr}} = \mathbf{H} \quad \delta\boldsymbol{x}_{\mathrm{p}}$	$\mathbf{J}_{\mathrm{c}} \quad \delta\boldsymbol{x}_{\mathrm{b}} = \delta\boldsymbol{x}_{\mathrm{p}}$
	$(6\times\boldsymbol{m})(\boldsymbol{m}\times 1) \quad (6\times 1)$	$(\boldsymbol{n}\times 6)(6\times 1) \quad (\boldsymbol{n}\times 1) \quad (\boldsymbol{n}\times 6)(6\times 1)$	$(6\times 6)(6\times 1) \quad (6\times 1)$
	Joints $\rightleftharpoons$	**Contact** $\rightleftharpoons$	**Object**
Force	$\mathbf{J}_\theta^{\mathrm{T}} \quad \boldsymbol{f}_{\mathrm{f}} = \boldsymbol{\tau}$	$\boldsymbol{f}_{\mathrm{f}} = \mathbf{H}^{\mathrm{T}} \quad \boldsymbol{f}_{\mathrm{tr}} = \boldsymbol{f}_{\mathrm{p}}$	$\mathbf{J}_{\mathrm{c}}^{\mathrm{T}} \quad \boldsymbol{f}_{\mathrm{p}} = \boldsymbol{f}_{\mathrm{b}}$
	$(\boldsymbol{m}\times 6)(6\times 1) \quad (\boldsymbol{m}\times 1)$	$(6\times n)(n\times 1) \quad (6\times 1)$	$(6\times 6)(6\times 1) \quad (6\times 1)$

At each contact, the force and velocity constraints can be represented by an $n\times 6$ *constraint* or *selection* matrix, $\mathbf{H}$ [37.3], at the contact interface. This constraint matrix works like a filter to transmit or deny certain components of motion across the contact interface. By the same token, forces/moments applied across the contact interface are also filtered by the same constraint matrix, $\mathbf{H}^{\mathrm{T}}$, in a dual relationship. Three typical contact models are illustrated below. Contact forces and wrenches are measured in a frame at the contact, with the contact normal in the $+z$-direction:

- The *point contact without friction* model. Only a normal force f_z can be exerted between the contacting bodies

$$f_z \geq 0\,, \quad \boldsymbol{w} = \mathbf{H}^{\mathrm{T}} f_z = (0\,0\,1\,0\,0\,0)^{\mathrm{T}} f_z\,. \tag{37.10}$$

- The *point contact with friction* model includes tangential friction forces, f_x and f_y, in addition to normal force f_z,

$$f_z \geq 0\,; |f_t| = \sqrt{f_x^2 + f_y^2} \leq \mu f_z\,,$$

$$\boldsymbol{w} = \mathbf{H}^{\mathrm{T}} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix}, \tag{37.11}$$

where f_x and f_y are the x- and y-components of the tangential contact force, and f_z is the normal force.

- The *soft finger* contact model with a finite contact patch allows, in addition to the friction and normal forces, a torsional moment with respect to the contact normal [37.35–39]. We have

$$f_z \geq 0\,;$$

$$\boldsymbol{w} = \mathbf{H}^{\mathrm{T}} \begin{pmatrix} f_x \\ f_y \\ f_z \\ m_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ f_z \\ m_z \end{pmatrix}, \tag{37.12}$$

where m_z is the moment with respect to the normal of contact. The finite contact area assumed by the soft contact interface results in the application of a friction moment in addition to the traction forces. If the resultant force on the tangential plane of contact is denoted as

$$f_t = \sqrt{f_x^2 + f_y^2}$$

and the moment with respect to the contact normal is m_z, the following elliptical equation represents the relationship between the force and moment at the onset of sliding

$$\frac{f_t^2}{a^2} + \frac{m_z^2}{b^2} = 1\,, \tag{37.13}$$

where $a = \mu f_z$ is the maximum friction force, and $b = (m_z)_{\max}$ is the maximum moment defined in equation (37.36). Further readings on this subject can be found in [37.35–42].

As can be appreciated from the three cases above, the rows of $\mathbf{H}$ are the directions along which contact forces are supported. Conversely, relative motions of the two objects are constrained along these same directions: $\mathbf{H}\dot{\boldsymbol{q}} = 0$. Thus, the constraint matrix works like a kinematic filter which dictates the components

of motions that can be transmitted through the contact interface. The introduction of this matrix makes it easy to model the contact mechanics in analysis using the dual kinematic relationship between force/moment and motion, as described in Table 37.1. For example, the constraint filter matrix in (37.12) describes that all three components of forces and one component of the moment with respect to the normal of contact can be transmitted through the contact interface of a soft finger.

In Chap. 38, when multiple contacts are considered, such $\mathbf{H}$ matrices associated with each finger can be concatenated in an augmented matrix to work in conjunction with the grasp and Jacobian matrices for the analysis of grasping and manipulation.

Table 37.1 summarizes the dual relationship between the force/moment and displacement – a result that is derived from the principle of virtual work. In Table 37.1, $\mathbf{J}_\theta$ is the joint Jacobian matrix relating joint velocities to fingertip velocities, and $\mathbf{J}_c$ is the Cartesian coordinate transformation matrix relating the contact point with the centroidal coordinates of the grasped object. In addition, the number of degrees of freedom (DOFs) in the task space is n and the number of DOFs in the joint spaces is m.

37.4 Rigid-Body Mechanics with Friction

The manipulation planning problem is to choose the motions or forces applied by manipulator contacts so that the part (or parts) moves as desired. This requires solving the subproblem of determining the motion of parts given a particular manipulator action.

Let $\boldsymbol{q} \in \mathbb{R}^n$ be local coordinates describing the combined configuration of the system consisting of one or more parts and robot manipulators, and let $\boldsymbol{w}_i \in \mathbb{R}^6$ represent a wrench at an active contact i, measured in the common coordinate frame O. Let $\boldsymbol{w}_{\text{all}} \in \mathbb{R}^{6k}$ be the vector obtained by stacking the $\boldsymbol{w}_i$, $\boldsymbol{w}_{\text{all}} = (\boldsymbol{w}_1^{\mathrm{T}}, \boldsymbol{w}_2^{\mathrm{T}}, \ldots, \boldsymbol{w}_k^{\mathrm{T}})^{\mathrm{T}}$ (where there are k contacts), and let $\mathbf{A}(\boldsymbol{q}) \in \mathbb{R}^{n\times 6k}$ be a matrix indicating how (or if) each contact wrench acts on each part. (Note that a contact wrench $\boldsymbol{w}_i$ acting on one part means that there is a contact wrench $-\boldsymbol{w}_i$ acting on the other part in contact.) The problem is to find the contact forces $\boldsymbol{w}_{\text{all}}$ and the system acceleration $\ddot{\boldsymbol{q}}$, given the state of the system $(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the system mass matrix $\mathbf{M}(\boldsymbol{q})$ and resulting Coriolis matrix $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the gravitational forces $\mathbf{g}(\boldsymbol{q})$, the control forces $\boldsymbol{\tau}$, and the matrix $\mathbf{T}(\boldsymbol{q})$ indicating how the control forces $\boldsymbol{\tau}$ act on the system. (Alternatively, if we view the manipulators as position controlled, the manipulator components of $\ddot{\boldsymbol{q}}$ can be directly specified and the corresponding components of the control forces $\boldsymbol{\tau}$ solved for.) One way to solve this problem is to (a) enumerate the set of all possible contact modes for the current active contacts, and (b) for each contact mode, determine if there are wrenches $\boldsymbol{w}_{\text{all}}$ and accelerations $\ddot{\boldsymbol{q}}$ that satisfy the dynamics

$$\mathbf{A}(\boldsymbol{q})\boldsymbol{w}_{\text{all}} + \mathbf{T}(\boldsymbol{q})\boldsymbol{\tau} - \mathbf{g}(\boldsymbol{q}) = \mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \,, \tag{37.14}$$

and that are consistent with the contact mode's kinematic constraints (constraints on $\ddot{\boldsymbol{q}}$) and friction cone force constraints (constraints on $\boldsymbol{w}_{\text{all}}$). This formulation is quite general and applies to multiple parts in contact. Equation (37.14) may be simplified by appropriate representations of the configurations and velocities of rigid-body parts (for example, using angular velocities of the rigid bodies instead of derivatives of local angular coordinates).

This formulation leads to some surprising conclusions: there may be multiple solutions to a particular problem (ambiguity) or there may be no solutions (inconsistency) [37.7–14]. This strange behavior arises from the status of Coulomb's law as an approximate law, and it disappears for zero friction, or friction that is small enough. Despite this weakness of the Coulomb friction law, it is a useful approximation. Nonetheless, if we would like to prove (under the Coulomb model) that a particular desired part motion occurs, we generally must also show that no other motion can occur. Otherwise we have only shown that the desired motion is one of the possible outcomes.

37.4.1 Complementarity

The conditions that each contact provides an equal number of motion constraints and wrench freedoms (Sect. 37.3.2) can be written as *complementarity* conditions. Thus the problem of solving (37.14), subject to contact constraints, can be formulated as a complementarity problem (CP) [37.12, 13, 43–46]. For planar problems, or spatial problems with approximate pyramidal friction cones, the problem is a linear complementarity problem [37.47]. For circular spatial friction cones, the problem is a nonlinear complementarity problem, due to the quadratic constraints describing the cones. In either case, standard algorithms can be used to solve for possible contact modes and part motions.

Alternatively, assuming linearized friction cones, we can formulate a linear constraint satisfaction program (LCSP) for each contact mode (e.g., a linear program with no objective function). The contact mode places linear constraints on the part accelerations, and the solver solves for the part accelerations and the nonnegative coefficients multiplying each of the edges of the friction cones, subject to the dynamics (37.14). Each LCSP with a feasible solution represents a feasible contact mode.

37.4.2 Quasistatic Assumption

A common assumption in robot manipulation planning is the *quasistatic* assumption. This assumption says that parts move slowly enough that inertial effects are negligible. This means that the right- and left-hand sides of (37.14) are zero. With this assumption, we often solve for part *velocities* rather than accelerations. These velocities must be consistent with the kinematic constraints and force constraints, and forces acting on the parts must always sum to zero.

37.4.3 Examples

While rigid-body mechanics problems with friction are usually solved using computational tools for CPs and LCSPs, equivalent graphical methods can be used for some planar problems to assist the intuition. As a simple example, consider the pipe clamp in Fig. 37.6 [37.14]. Under the external wrench $\boldsymbol{w}_{\text{ext}}$, does the clamp slide down the pipe, or does it remain fixed in place? The figure uses moment labeling to represent the composite wrench cone of contact forces that can act on the clamp from the pipe. The wrench $\boldsymbol{w}_{\text{ext}}$ acting on the clamp can be exactly balanced by a wrench in the composite wrench cone. This is evident from the fact that the wrench opposing $\boldsymbol{w}_{\text{ext}}$ passes around the + labeled region in a counterclockwise sense, meaning that it is contained in the contact wrench cone. As a result, static equilibrium (the `ff` contact mode) is a feasible solution for the pipe clamp. To arrive at the same result using an LCSP, label the unit wrenches at the edges of the friction cones $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_4$. Then the clamp can remain at rest if there exist coefficients $a_1, \ldots, a_4$ such that

$$a_1, a_2, a_3, a_4 \geq 0 \,,$$
$$a_1\boldsymbol{w}_1 + a_2\boldsymbol{w}_2 + a_3\boldsymbol{w}_3 + a_4\boldsymbol{w}_4 + \boldsymbol{w}_{\text{ext}} = 0 \,.$$

To show that static equilibrium is the only solution, all other contact modes must be ruled out. Note that, if the friction coefficient at the contacts is too small, the clamp will fall under $\boldsymbol{w}_{\text{ext}}$.

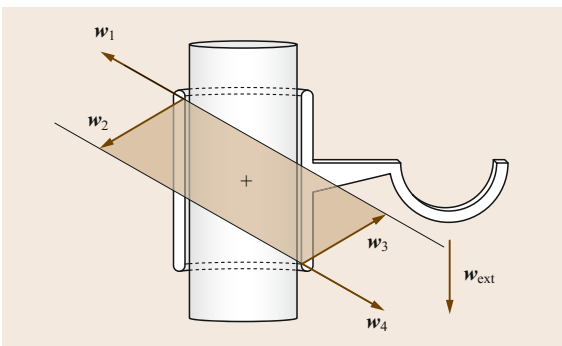


Fig. 37.6 The moment-labeling representation of the composite wrench cone of contact forces the pipe can apply to the pipe clamp. In this figure, the wrench $\boldsymbol{w}_{\text{ext}}$ applied to the pipe clamp can be resisted by forces within the composite wrench cone

Analysis of the classic peg-in-hole problem is similar to that of the pipe clamp. Figure 37.7a shows an angled peg making two-point contact with the hole. If we apply the wrench $\boldsymbol{w}_1$, the forces from the two-point contact cannot resist, so the `ff` mode contact mode is not a possibility, and the peg will continue to move into the hole. If we apply the wrench $\boldsymbol{w}_2$, however, the contacts can resist, and the peg will get stuck. This is called *jamming* [37.48]. If there is higher friction at the contacts, as in Fig. 37.7b, then each friction cone may be able to see the base of the other, and the feasible contact wrenches span the entire wrench space [37.49]. In this case, any wrench we apply may be resisted by the contacts. The peg is said to be *wedged* [37.48]. Whether or not the peg will actually resist our applied wrench depends on how much internal force acts between the contacts. This cannot be answered by our rigid-body model; it requires a compliant contact model, as discussed in Sect. 37.6.

As a slightly more complex quasistatic example, consider a block on a table being pushed by a finger

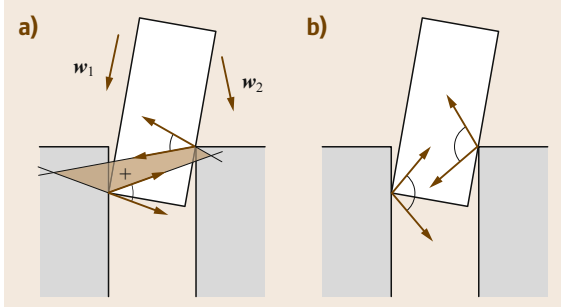


Fig. 37.7 **(a)** The peg will proceed into the hole under the external wrench $\boldsymbol{w}_1$ but will get stuck under the wrench w_2. **(b)** The peg is wedged

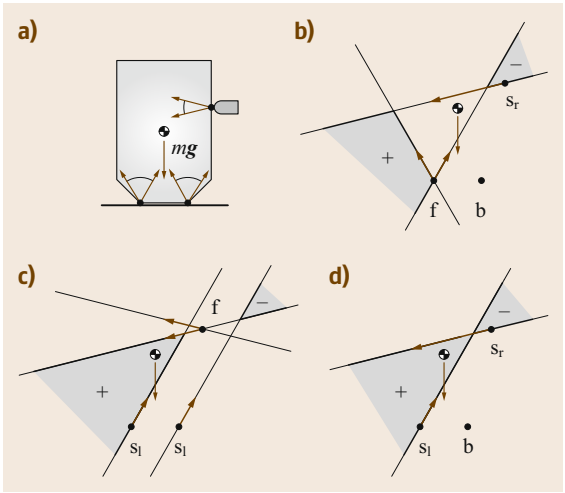


Fig. 37.8 **(a)** A finger moves *left* into a planar block resting on a table in gravity. The contact friction cones are shown. **(b)** Possible contact forces for the contact label f at the leftmost bottom contact, b at the rightmost bottom contact, and s_r at the pushing contact. This contact mode fbr corresponds to the block tipping over the leftmost contact. Note that the contact wrench cone (represented by moment labels) can provide a force that exactly balances the gravitational force. Thus this contact mode is quasistatically possible. **(c)** The contact wrench cone for the $s_l s_l f$ contact mode (block slides *left* on the table) cannot balance the gravitational force. This contact mode is not quasistatically possible. **(d)** The contact wrench cone for the $s_l b s_r$ contact mode (block slides and tips) is not quasistatically possible

that moves horizontally to the left (Fig. 37.8) [37.50, 51]. Does the part begin to tip over, slide, or both tip and slide? The corresponding contact modes are fbs_r, $s_l s_l f$, and $s_l b s_r$, as shown in Fig. 37.8. The figure also shows the composite contact wrench cones using moment labels. It is clear that quasistatic balance between the gravitational force and the contact wrench cone can only occur for the tipping without slipping contact mode fbs_r. Therefore, the only quasistatic solution is that the block begins to tip without sliding, and the speed of this motion is determined by the speed of the finger's motion. The graphical construction can be used to confirm our intuition that tipping of the block occurs if we push high on the block or if the friction coefficient at the block's support is high. Try it by pushing a can or

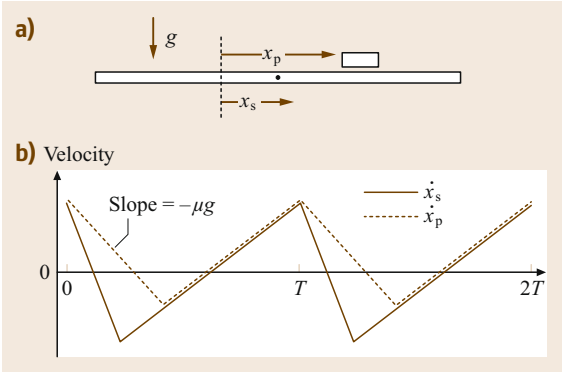


Fig. 37.9 **(a)** A part supported by a horizontally vibrating surface. **(b)** Friction between the part and the surface causes the part to always try to catch up to the surface, but its acceleration is bounded by $\pm\mu g$. The asymmetric motion of the surface gives the part an average positive velocity over a cycle

glass. Quasistatically, the height of the center of mass is immaterial.

A final example is given in Fig. 37.9. A part of mass m and friction coefficient μ is supported by a horizontal surface that moves periodically in the horizontal direction. The periodic motion consists of a large negative acceleration for a short duration and a smaller positive acceleration for a longer duration. The horizontal friction force that can be applied to the part is bounded by $\pm\mu mg$, so the part's horizontal acceleration is bounded by $\pm\mu g$. As a result, the part cannot keep up with the surface as it accelerates backward. Instead, the part slips forward with respect to the surface, which means that the maximal friction force acts in the negative direction, and the part attempts to slow down to the surface's velocity. Eventually the part catches up with the surface as it executes its slow forward acceleration, which is less than μg. Once the part catches up to the surface, it sticks to it, until the next backward acceleration phase. As is evident in Fig. 37.9, the average part velocity over a cycle is positive, so the part moves forward on the surface [37.52, 53]. This idea can be extended to create a wide variety of frictional force fields on a rigid plate vibrated with three degrees of freedom in a horizontal plane [37.54, 55] or with a full six degrees of freedom [37.56].

For further examples of manipulation planning using the rigid-body model with Coulomb friction, see [37.14] and the references therein.

Avidbots Ex1016
Page 973 of 2154

37.5 Pushing Manipulation

The friction limit surface (Sect. 37.7) is useful for analyzing pushing manipulation, as it describes the friction forces that can occur as a part slides over a support surface. When the part is pushed with a wrench contained within the limit surface, friction between the part and the support resists the pushing wrench and the part remains motionless. When the part slides quasistatically, the pushing wrench $\boldsymbol{w}$ lies on the limit surface, and the part's twist $\boldsymbol{t}$ is normal to the limit surface at $\boldsymbol{w}$ (Fig. 37.10). When the part translates without rotating, the friction force magnitude is μmg, where m is the part's mass and g is gravitational acceleration. The force applied by the part to the surface is directed through the part's center of mass in the direction of translation.

If the pushing wrench makes positive moment about the part's center of mass, the part will rotate counterclockwise (CCW), and if it makes negative moment about the part's center of mass, it will rotate clockwise (CW). Similarly, if the contact point on the part moves along a line that passes around the part's center of mass in a CW (respectively, CCW) sense, then the part will rotate CW (CCW). From these two observations and considering all possible contact modes, we can conclude that a part pushed at a point contact will rotate CW (CCW) if either (1) both edges of the contact friction cone pass CW (CCW) about the center of mass, or (2) one edge of the friction cone and the pushing direction at the contact both pass around the center of mass in a CW (CCW) sense [37.14, 57], (Fig. 37.11).

This observation allows pushing to be used to reduce uncertainty in part orientation. A series of pushes with a flat fence can be used to completely eliminate the uncertainty in the orientation of a polygonal part [37.57–59]. Bounds on the rate of rotation of parts [37.14, 60, 61] allow the use of a sequence of stationary fences suspended above a conveyor belt to orient parts by pushing them as they are carried along by the conveyor [37.62, 63]. *Stable* pushing plans (Fig. 37.12) use pushing motions that are guaranteed to keep the part fixed to the pusher as it moves, even

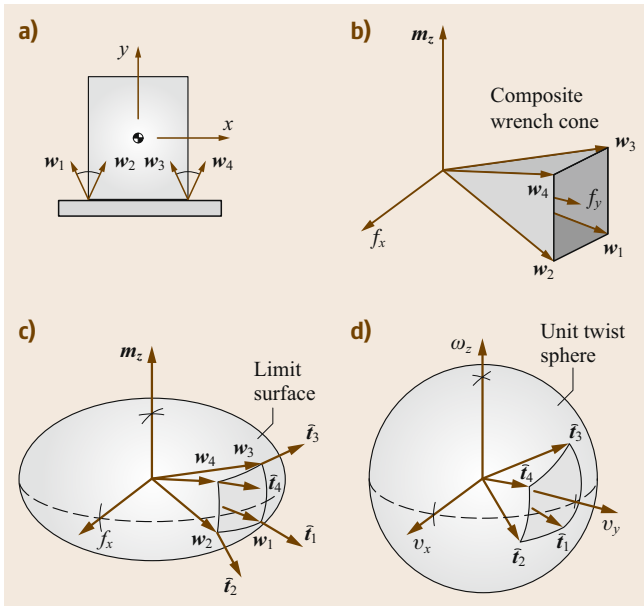


Fig. 37.10 (a) Contact between a pusher and a sliding object. (b) The composite contact wrench cone. (c) Mapping the wrench cone through the limit surface to twists of the object. (d) The unit twists that can result from forces in the composite wrench cone

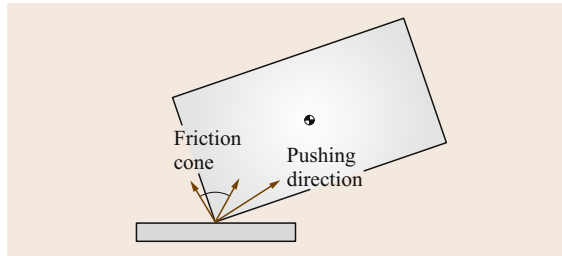


Fig. 37.11 The pusher's motion direction votes for counterclockwise rotation of the part, but it is outvoted by the two edges of the friction cone, which both indicate clockwise rotation

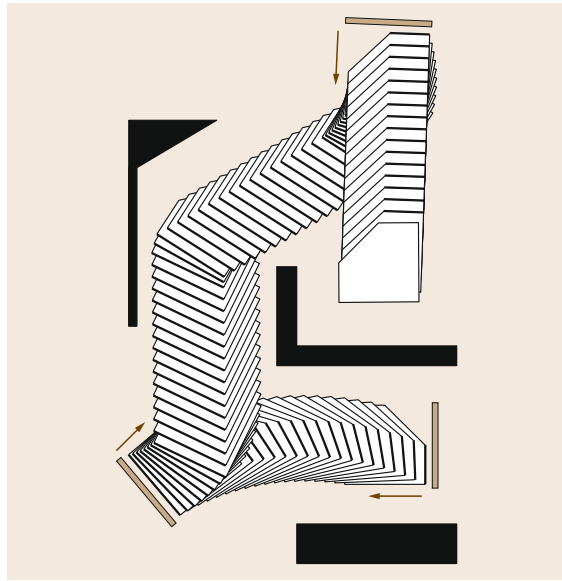


Fig. 37.12 Stable pushes can be used to maneuver a part among obstacles

in the face of uncertainty in the part's pressure distribution [37.14, 64, 65]. Extensions of this work find pushing motions for planar assemblies of parts so that they remain fixed in their relative configurations during motion [37.66–68].

The examples above assume that pushing forces and support friction forces act in the same plane. Other work on pushing has considered three-dimensional effects, where pushing forces are applied above the support plane [37.69].

37.6 Contact Interfaces and Modeling

Contact interface is a general expression to describe the kinematics and kinetics of contacts. Contact applies in various contexts in robotics research; thus, it is more meaningful to refer to generic contact as an *interface*, which is not limited to fingers in grasping and manipulation. The concept of *contact interface* extends the traditional context of physical *contact*. It refers to an interface which imparts kinematic filtering as well as the duality of force/motion transmitted across the contact interface.

Consequently, a contact interface, whether it is rigid or deformable, can be considered as a *kinematic filter* which includes two characteristics: (a) motion and force transmission, and (b) kinematic duality between force/moment and motion. Different contact interfaces will be described in the following sections.

Previous sections of this chapter have assumed rigid bodies in contact. In reality, however, all contacts are accompanied by some deformation of the objects. Often this is by design, as in the case of compliant robot fingertips. When deformation is non-negligible, elastic contact models can be used.

37.6.1 Modeling of Contact Interface

Contact modeling depends on the nature of bodies in contact, including their material properties, applied force, contact deformation, and elastic properties. This section discusses different contact models.

Rigid-Body and Point-Contact Models

With the rigid-body assumption, as discussed in the previous sections of this chapter, two models are often used: (a) *point contact without friction* and (b) *point contact with friction*. In the former case, the contact can only apply force in the direction normal to the contact. In the latter case, a tangential friction force is applied in addition to the normal force. The simplest analytical model for point contact with friction is the Coulomb friction model as presented in (37.9).

Hertzian Contact Model

Elastic contact modeling was first studied and formulated more than a century ago by *Hertz* in 1882 [37.70] based on contact between two linear elastic materials with a normal force which results in very small contact deformation. This is commonly called the *Hertzian* contact and can be found in most mechanics textbooks, such as [37.71, 72]. Hertz made two important explicit assumptions in order for his contact model to be applicable:

1. Objects of linear elastic materials in contact,
2. Small contact deformation compared to the dimension of object.

Hertz also conducted experiments using a spherical glass lens against a planar glass plate to validate the contact theory.

Two relevant results of the Hertzian contact theory applied to robotic contact interfaces are summarized as follows. The first pertains to the radius of the contact area. *Hertz* [37.70] studied the growth of the contact area as a function of the applied normal force N based on the linear elastic model. Based on ten experimental trials, he concluded that the radius of contact is proportional to the normal force raised to the power of $1/3$, which is consistent with the analytical results he derived based on the linear elastic model. That is, the radius of contact, a, is related to the normal force, N, by

$$a \propto N^{\frac{1}{3}} \,. \tag{37.15}$$

The second result pertains to the pressure distribution over the assumed symmetric contact area – a second-order pressure distribution of the nature of an ellipse or circle. For a symmetric and circular contact area, the pressure distribution is

$$p(r) = \frac{N}{\pi a^2}\sqrt{1-\left(\frac{r}{a}\right)^2}, \tag{37.16}$$

where N is the normal force, a is the radius of contact, and r is the distance from the center of contact with $0 \le r \le a$.

Soft Contact Model. A typical contact interface between a soft finger and contact surface is illustrated in Fig. 37.13. In typical robotic contact interfaces, the materials of the fingertips are not linear elastic. A model

that extends linear to nonlinear elastic contact was presented in [37.38] with a power-law equation which subsumes the Hertzian contact theory,

$$a = c\,N^{\gamma}\,, \tag{37.17}$$

where $\gamma = n/(2n+1)$ is the exponent of the normal force, n is the strain-hardening exponent, and c is a constant depending on the size and curvature of the fingertip as well as the material properties. Equation (37.17) is the new power law that relates the growth of the circular contact radius to the applied normal force for soft fingers. Note that the equation is derived assuming circular contact area. For linear elastic materials, the constant n is equal to 1 with $\gamma = 1/3$, resulting in the Hertzian contact model in (37.15). Therefore, the soft contact model in (37.17) subsumes the Hertzian contact model.

Viscoelastic Soft Contact Model. Typical soft materials used in robotic fingertip, e.g., rubber, silicone and polymers, show viscoelastic properties. Viscoelasticity is a physical phenomenon of time-dependent strain and stress [37.73]. Specifically in the context of robotic grasping and manipulation, there are two types of time dependent responses [37.74–76]:

1. *Relaxation*: The evolution of force in grasping while the displacement is held constant;
2. *Creep*: The evolution of displacement in contact and grasping while the external force is held constant.

Such time-dependent responses would approach equilibrium asymptotically. Viscoelastic materials also exhibit the properties of:

1. *Strain history dependence*: The response of the material depends on prior strain history; and
2. *Energy dissipation*: A net energy dissipation associated with a complete cycle of loading and unloading.

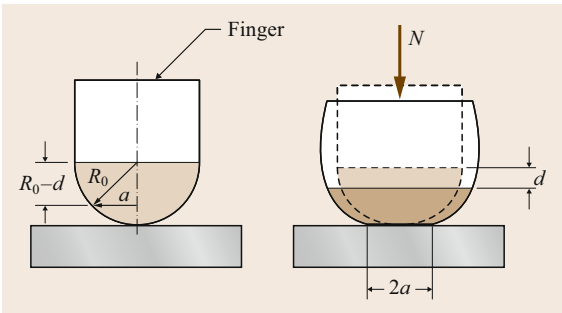


Fig. 37.13 An elastic soft fingertip with a hemispherical tip making contact with a rigid surface

Two popular viscoelastic models include: the Kelvin–Voigt/Maxwell model (the spring-damper model) and the Fung's model, which will be described in the following.

(1) Kelvin–Voigt/Maxwell model. The Kelvin–Voigt's solid model was constructed by a spring and a damper connected in parallel [37.77] as shown in Fig. 37.14. The relation between the stress, σ, and strain, ϵ, can be formulated as

$$\sigma = k\epsilon + c\frac{\mathrm{d}\epsilon}{\mathrm{d}t}\,. \tag{37.18}$$

Maxwell proposed in 1867 the Maxwell fluid model [37.78] as a single set of spring and damper in series as shown in Fig. 37.15. The relationship between the stress, σ, and strain, ϵ, can be expressed as

$$\frac{1}{k}\frac{\mathrm{d}\sigma}{\mathrm{d}t} + \frac{\sigma}{c} = \frac{\mathrm{d}\epsilon}{\mathrm{d}t}\,. \tag{37.19}$$

The *generalized Maxwell model* utilizes multiple serial spring-damper sets and a spring connected in parallel. One can employ the curve-fitting techniques using the experimental data to find the modeling parameters of springs and dampers.

However, the issue of this model is often on the lack of consistency of the parameters (stiffness constants and damping factors) obtained from the model. Such parameters can have large discrepancy in numerical values, often in two orders of magnitude or higher, while representing the same material with supposedly similar values of springs and dampers. These parameters sometimes can also present unrealistic difference in scales under different experimental setup (e.g., in [37.79, 80]).

(2) Fung's model. A popular viscoelastic model in modeling biomedical materials is the Fung's model [37.76], proposed by *Fung* in 1993. The main idea of the model is to represent the reacting force as the product of two independent responses: the temporal response and the elastic response, while incorporating

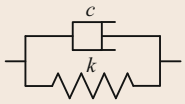


Fig. 37.14 Kelvin–Voigt model

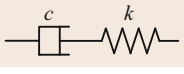


Fig. 37.15 Maxwell model

the history of the stress response. The model may be written as

$$T(t) = \int_{-\infty}^{t} G(t-\tau)\frac{\partial T^{(e)}[\lambda(t)]}{\partial \lambda}\frac{\partial \lambda(\tau)}{\partial \tau}\,\mathrm{d}\tau\,, \quad (37.20)$$

where $T(t)$ is the tensile stress at time t, with a step increase of size λ in elongation on the specimen, the function $T^{(e)}(\lambda)$ is the so-called elastic response, and $G(t)$, a normalized function of time, is the reduced relaxation function.

Tiezzi and *Kao* [37.42, 81, 82] simplified this model to study the soft contact interface by assuming no past stress history, as expressed in (37.21)

$$\mathcal{G}(\delta, t) = N^{(e)}(\delta)\cdot g(t)\,, \quad (37.21)$$

where $\mathcal{G}(\delta, t)$ represents the grasping force as a function of the displacement δ and time t, $N^{(e)}(\delta)$ represents the elastic response of normal force as a function of the displacement (or depression unto the object), and $g(t)$ represents the temporal response of relaxation or creep. The important property of this model is the separation of the spatial response and temporal response as two independent functions. It was shown that the viscoelastic model has an important implication on the stability of grasping which cannot be captured by rigid or linear elastic modeling.

Other Models. In addition to the aforementioned models, other models from different viewpoints were proposed, as follows: From the viewpoint of rheological perspectives [37.83–86]; molecular perspectives [37.73, 87–92]; energy perspectives [37.85]; distributed modeling perspectives [37.93–98] and stress wave propagation perspectives [37.99–102].

37.6.2 Pressure Distribution at Contacts

In Sect. 37.6.1, when the Hertzian contact theory is considered, the assumed pressure distribution for small elastic deformation is given in (37.16). As the radius of curvature of the two asperities increases and the material properties change to hyperelastic, the pressure distribution becomes more uniform [37.38, 103, 104]. Generalizing equation (37.16), the pressure distribution function for circular contact area with radius a is

$$p(r) = C_k\frac{N}{\pi a^2}\left[1-\left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}}\,, \quad (37.22)$$

where N is the normal force, a is the radius of contact, r is the radius with $0 \le r \le a$, k determines the shape of the pressure profile, and C_k is a coefficient that adjusts for the profile of pressure distribution over the contact area to satisfy the equilibrium condition. In (37.22), $p(r)$ is defined for $0 \le r \le a$. By symmetry, $p(r) = p(-r)$ when $-a \le r \le 0$, as shown in Fig. 37.16. When k becomes larger, the pressure distribution approaches uniform distribution, as shown in Fig. 37.16. It is also required that the integral of the pressure over the contact area be equal to the normal force; that is

$$\int_R p(r)\,\mathrm{d}A = \int_{\theta=0}^{2\pi}\int_{r=0}^{a} p(r)\,r\,\mathrm{d}r\,\mathrm{d}\theta = N\,. \quad (37.23)$$

The coefficient C_k can be obtained by substituting (37.22) into (37.23). It is interesting to note that when (37.23) is integrated, both the normal force and radius of contact vanish, leaving only the constant C_k as follows

$$C_k = \frac{3}{2}\frac{k\Gamma\left(\frac{3}{k}\right)}{\Gamma\left(\frac{1}{k}\right)\Gamma\left(\frac{2}{k}\right)}\,, \quad (37.24)$$

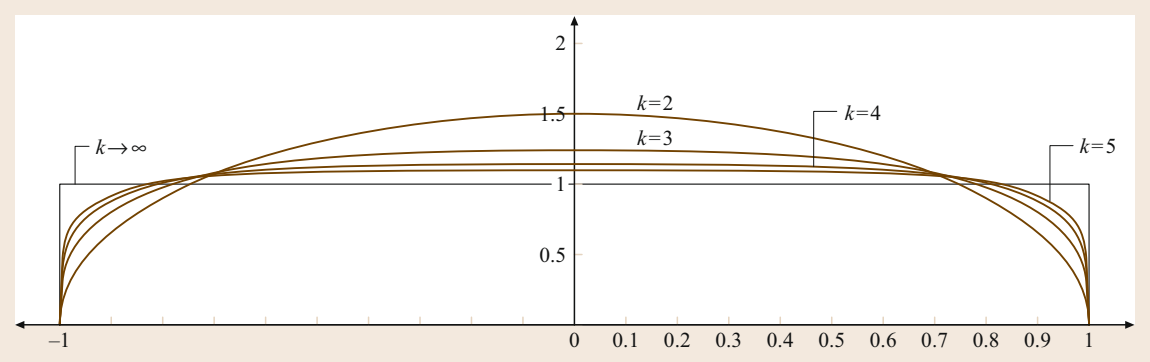


Fig. 37.16 Illustration of pressure distribution with respect to the normalized radius, r/a. The plot shows an axisymmetric pressure distribution with $k = 2, 3, 4, 5$, and ∞. As k increases, the pressure profile becomes more uniform

Avidbots Ex1016
Page 977 of 2154

where $k = 1, 2, 3, \ldots$ are typically integer values (although noninteger k values are also possible) and $\Gamma()$ is the gamma function [37.105]. The numerical values of C_k for a few values of k are listed in Table 37.2 for reference. A normalized pressure distribution with respect to the normalized radius, r/a, is plotted in Fig. 37.16. As can be seen from the figure, when k approaches infinity, the pressure profile will become that of a uniformly distributed load with magnitude of $N/(\pi a^2)$, in which case $C_k = 1.0$.

For linear elastic materials, $k = 2$ can be used, although it is found that $k \cong 1.8$ is also appropriate in some cases [37.106]. For nonlinear elastic and viscoelastic materials, the value of k tends to be higher, depending on the properties of the material.

Table 37.2 Values of the coefficient C_k in the pressure distribution (37.22)

k	Coefficient C_k
$k = 2$ (circular)	$C_2 = 1.5$
$k = 3$ (cubic)	$C_3 = 1.24$
$k = 4$ (quadruple)	$C_4 = 1.1441$
$k \to \infty$ (uniform)	$C_\infty = 1.0$

37.7 Friction Limit Surface

Section 37.3 introduced the notion of a wrench cone, describing the set of wrenches that can be applied through a point contact with friction. Any contact wrench is limited to lie within the surface of the cone. This gives rise to the general notion of a *friction limit surface* – the surface bounding the set of wrenches that can be applied through a given contact or set of contacts.

In this section we study the particular case of limit surfaces arising from planar contact patches [37.7, 107]. A planar contact patch occurs when a flat object slides on the floor, a soft robot finger presses against a face of a polyhedron, or a foot of a humanoid robot pushes on the ground. We would like to know what wrenches can be transmitted through such a contact. In the rest of this section, we focus on the properties of the limit surface of a planar contact patch with a specified pressure distribution over the patch.

For ease of discussion, we will call one of the objects in contact the *part* (e.g., the flat object on the floor or the robot finger) and the other object a stationary *support*. We define a coordinate frame so that the planar contact patch is in the $z = 0$ plane, and let $p(\boldsymbol{r}) \geq 0$ be the contact pressure distribution between the part and support as a function of the location $\boldsymbol{r} = (x, y)^{\mathrm{T}}$. The friction coefficient at the contact patch is μ. If the planar velocity of the part is $\boldsymbol{t} = (\omega_z, v_x, v_y)^{\mathrm{T}}$, then the linear velocity at $\boldsymbol{r}$ is

$$\boldsymbol{v}(\boldsymbol{r}) = (v_x - \omega_z y, v_y + \omega_z x)^{\mathrm{T}} ,$$

and the unit velocity is $\hat{\boldsymbol{v}}(\boldsymbol{r}) = \boldsymbol{v}(\boldsymbol{r})/\|\boldsymbol{v}(\boldsymbol{r})\|$. The infinitesimal force applied by the part to the support at $\boldsymbol{r}$, in the plane of sliding, is

$$\mathrm{d}\boldsymbol{f}(\boldsymbol{r}) = [\mathrm{d}f_x(\boldsymbol{r}), \mathrm{d}f_y(\boldsymbol{r})]^{\mathrm{T}} = \mu p(\boldsymbol{r}) \hat{\boldsymbol{v}}(\boldsymbol{r}) . \qquad (37.25)$$

The total wrench the part applies to the support is

$$\boldsymbol{w} = \begin{pmatrix} m_z \\ f_x \\ f_y \end{pmatrix} = \int_A \begin{pmatrix} x\,\mathrm{d}f_y(\boldsymbol{r}) - y\,\mathrm{d}f_x(\boldsymbol{r}) \\ \mathrm{d}f_x(\boldsymbol{r}) \\ \mathrm{d}f_y(\boldsymbol{r}) \end{pmatrix} \mathrm{d}A , \qquad (37.26)$$

where A is the support area.

As expected, the wrench is independent of the speed of motion: for a given $\boldsymbol{t}_0$ and all $\alpha > 0$, the wrench resulting from velocities $\alpha \boldsymbol{t}_0$ is identical. Viewing (37.26) as a mapping from twists to wrenches, we can map the surface of the unit twist sphere ($\|\hat{\boldsymbol{t}}\| = 1$) to a surface in the wrench space. This surface is the limit surface, and it is closed, convex, and encloses the origin of the wrench space (Fig. 37.17). The portion of the wrench

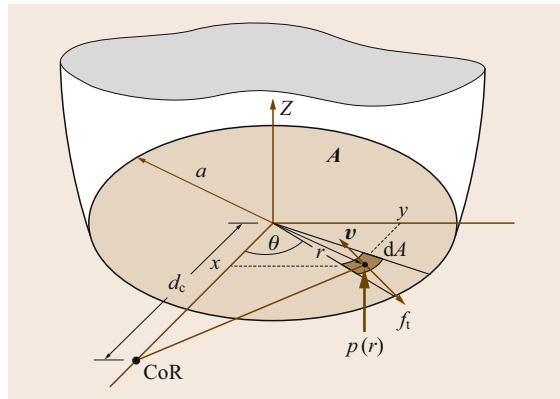


Fig. 37.17 Contact and coordinates for COR and local infinitesimal area $\mathrm{d}A$ for numerical integration to construct the limit surface of soft fingers

space enclosed by this surface is exactly the set of wrenches the part can transmit to the support. When the part slips on the support ($\boldsymbol{t} \neq 0$), the contact wrench $\boldsymbol{w}$ lies on the limit surface, and by the maximum-work inequality, the twist $\boldsymbol{t}$ is normal to the limit surface at $\boldsymbol{w}$. If the pressure distribution $p(\boldsymbol{r})$ is finite everywhere, the limit surface is smooth and strictly convex, and the mapping from unit twists to unit wrenches, and vice versa, is continuous and one-to-one. The limit surface also satisfies the property $\boldsymbol{w}(-\hat{\boldsymbol{t}}) = -\boldsymbol{w}(\hat{\boldsymbol{t}})$.

37.7.1 The Friction Limit Surface at a Soft Contact Interface

Given a pressure distribution in (37.22) we can numerically construct the corresponding friction limit surface by using (37.25) and (37.26). For an infinitesimal contact area, the contact resembles a point contact. Hence, the Coulomb's law of friction can be employed.

In the following, we will exploit the kinematic property of center of rotation (COR) with modeling symmetry to formulate equations for the magnitudes of friction force and moment on a circular contact patch. Figure 37.17 shows a circular contact patch with an instantaneous COR. By moving (or scanning) the COR along the x-axis, different possible combinations of friction force and moment can be obtained to construct the limit surface. More details can be found in [37.35–38, 41, 41, 42, 107–109].

The following derivation is for the total friction force (f_t) and moment (m_z) on the contact interface for the COR at a distance d_c along the x-axis, as shown in Fig. 37.17. By varying the COR distance d_c from $-\infty$ to ∞, all possible combinations of (f_t, m_z) can be found in order to construct the entire friction limit surface.

The tangential force over the entire contact area can be obtained by integrating shear force on each infinitesimal areas, $\mathrm{d}A$, on which the Coulomb's law of friction is observed, over the entire contact area, A. When Hertzian contact pressure distribution [37.70] is considered, use $k = 2$ in (37.22). Setting $\boldsymbol{r} = (x\,, y)^{\mathrm{T}}$ and $r = \|\boldsymbol{r}\|$, the total tangential force can be integrated using (37.25) to obtain

$$\boldsymbol{f}_t = \begin{pmatrix} f_x \\ f_y \end{pmatrix} = -\int_A \mu\, \hat{\boldsymbol{v}}(\boldsymbol{r})\, p(r)\, \mathrm{d}A\,, \tag{37.27}$$

where A denotes the circular contact region in Fig. 37.17, $\boldsymbol{f}_t$ is the tangential force vector with the direction shown in Fig. 37.17, μ is the coefficient of friction, $\hat{\boldsymbol{v}}(\boldsymbol{r})$ is the unit vector in the direction of the velocity vector $\boldsymbol{v}(\boldsymbol{r})$ with respect to the COR on the infinitesimal area $\mathrm{d}A$ at the location $\boldsymbol{r}$, and $p(r)$ is the pressure distribution at distance r from the center of contact. Since the pressure along the annular ring with distance r from the center is the same, we denote it by $p(r)$ instead of $p(\boldsymbol{r})$. The minus sign denotes the opposite directions of $\hat{\boldsymbol{v}}(\boldsymbol{r})$ and $\boldsymbol{f}_t$. Since we are primarily interested in the magnitude of the friction force and moment, we shall omit this sign in the later derivation when magnitudes are concerned.

Similarly, the moment about the z-axis, or the normal to the contact area, is

$$m_z = \int_A \mu \|\boldsymbol{r} \times \hat{\boldsymbol{v}}(\boldsymbol{r})\|\, p(r)\, \mathrm{d}A\,, \tag{37.28}$$

where $\|\boldsymbol{r} \times \hat{\boldsymbol{v}}(\boldsymbol{r})\|$ is the magnitude of the cross product of the vectors $\boldsymbol{r}$ and $\hat{\boldsymbol{v}}(\boldsymbol{r})$, whose direction is normal to the contact surface.

The unit vector $\hat{\boldsymbol{v}}(\boldsymbol{r})$ is related to the distance d_c from the origin to the COR chosen on the x-axis from Fig. 37.17, and can be written as follows

$$\begin{aligned} \hat{\boldsymbol{v}}(\boldsymbol{r}) &= \frac{1}{\sqrt{(x-d_c)^2 + y^2}} \begin{bmatrix} -y \\ (x-d_c) \end{bmatrix} \\ &= \frac{1}{\sqrt{(r\cos\theta - d_c)^2 + (r\sin\theta)^2}} \\ &\quad \times \begin{bmatrix} -r\sin\theta \\ (r\cos\theta - d_c) \end{bmatrix}. \end{aligned} \tag{37.29}$$

Due to symmetry, $f_x = 0$ for all CORs along the x-axis; therefore, the magnitude of the tangential force in the contact tangent plane is $f_t = f_y$. Substituting equations (37.22) and (37.29) into (37.27) and (37.28), we obtain

$$\begin{aligned} f_t &= \int_A \mu \frac{(r\,\cos\theta - d_c)}{\sqrt{r^2 + d_c^2 - 2rd_c\,\cos\theta}} C_k \\ &\quad \times \frac{N}{\pi a^2}\left[1 - \left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}} \mathrm{d}A\,. \end{aligned} \tag{37.30}$$

Similarly, the moment about the axis normal to the plane is

$$\begin{aligned} m_z &= \int_A \mu \frac{r^2 - r\,d_c\cos\theta}{\sqrt{r^2 + d_c^2 - 2rd_c\,\cos\theta}} C_k \\ &\quad \times \frac{N}{\pi a^2}\left[1 - \left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}} \mathrm{d}A. \end{aligned} \tag{37.31}$$

In (37.30) and (37.31), polar coordinates are defined such that $x = r\,\cos\theta$, $y = r\,\sin\theta$, and $\mathrm{d}A = r\,\mathrm{d}r\,\mathrm{d}\theta$. We also introduce a normalized coordinate

$$\tilde{r} = \frac{r}{a}\,. \tag{37.32}$$

Avidbots Ex1016
Page 979 of 2154

From (37.32), we can write $\mathrm{d}r = a\,\mathrm{d}\tilde{r}$. We also denote $\tilde{d}_\mathrm{c} = d_\mathrm{c}/a$, and assume that μ is constant throughout the contact area. Substituting the normalized coordinate into equation (37.30) and dividing both sides by μN, we can derive

$$\frac{f_\mathrm{t}}{\mu N} = \frac{C_k}{\pi}\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^2\cos\theta - \tilde{r}\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^k)^{\frac{1}{k}}\,\mathrm{d}\tilde{r}\,\mathrm{d}\theta\,. \tag{37.33}$$

Substituting again $\tilde{r} = r/a$ and $\mathrm{d}r = a\,\mathrm{d}\tilde{r}$ into (37.31) and normalizing with $a\mu N$, we obtain

$$\frac{m_z}{a\mu N} = \frac{C_k}{\pi}\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^3\cos\theta - \tilde{r}^2\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^k)^{\frac{1}{k}}\,\mathrm{d}\tilde{r}\,\mathrm{d}\theta\,. \tag{37.34}$$

Equations (37.33) and (37.34) can be numerically integrated for a distance d_c or $\tilde{d}_\mathrm{c}$ to yield a point on the limit surface for a prescribed pressure distribution $p(r)$ given by (37.22). Both equations involve elliptic integrals whose closed-form solutions may not exist but can be evaluated numerically. When the COR distance d_c varies from $-\infty$ to ∞, all possible combinations of (f_t, m_z) can be obtained for plotting the friction limit surface.

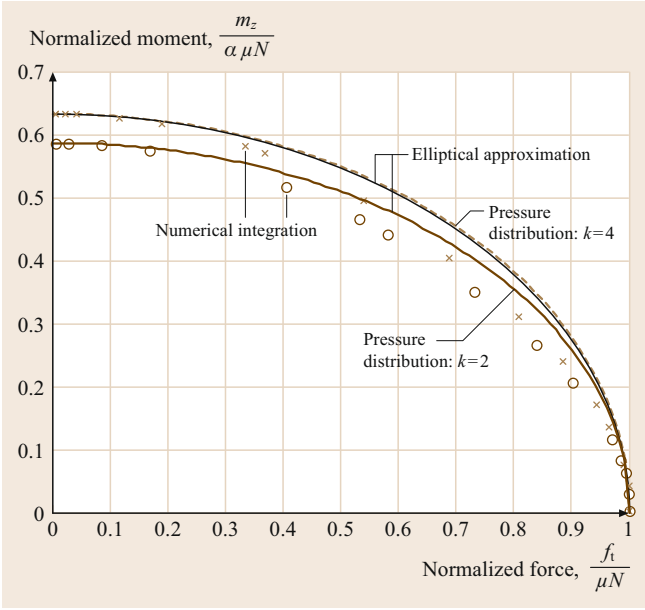


Fig. 37.18 Example limit surface obtained by numerical integration and elliptical approximation. The numerical integration is based on the pressure distribution, $p(r)$, and the coefficient, C_k. In this figure, the pressure distributions in (37.22) of both $k = 2$ and $k = 4$ are used

37.7.2 Example of Constructing a Friction Limit Surface

When the pressure distribution is fourth order with $k = 4$, the coefficient from Table 37.2 is $C_4 = (6/\sqrt{\pi})(\Gamma(3/4)/\Gamma(1/4)) = 1.1441$. Equations (37.33) and (37.34) can be written as

$$\frac{f_\mathrm{t}}{\mu N} = 0.3642\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^2\cos\theta - \tilde{r}\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^4)^{\frac{1}{4}}\,\mathrm{d}\tilde{r}\,\mathrm{d}\theta\,.$$

$$\frac{m_z}{a\mu N} = 0.3642\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^3\cos\theta - \tilde{r}^2\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^4)^{\frac{1}{4}}\,\mathrm{d}\tilde{r}\,\mathrm{d}\theta\,.$$

Numerical integration for different values of $\tilde{d}_\mathrm{c}$ yields pairs of $(f_\mathrm{t}/(\mu N),\ m_z/(a\mu N))$. Plots of these pairs are shown in Fig. 37.18.

A reasonable approximation to these numerical results is given by the following equation of an ellipse

$$\left(\frac{f_\mathrm{t}}{\mu N}\right)^2 + \left(\frac{m_z}{(m_z)_\mathrm{max}}\right)^2 = 1\,, \tag{37.35}$$

where the maximum moment $(m_z)_\mathrm{max}$ is

$$(m_z)_\mathrm{max} = \int_A \mu\,|r| C_k \frac{N}{\pi a^2}\left[1 - \left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}}\mathrm{d}A\,, \tag{37.36}$$

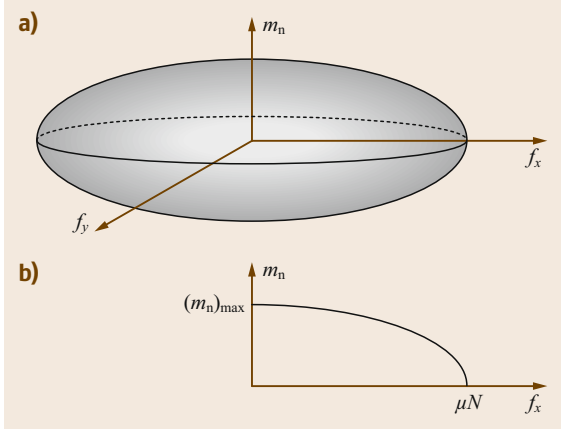


Fig. 37.19a,b Friction limit surface for soft fingers: **(a)** a 3-D ellipsoid representing the limit surface, and **(b)** a section of the ellipsoidal limit surface showing the coupled relationship between the force and moment, as those obtained in Fig. 37.18

obtained by (37.31) with the COR at $d_c = 0$. This defines the quarter-elliptical curves in Fig. 37.18. This approximation is the basis for constructing a three-dimensional (3-D) ellipsoidal limit surface, as illustrated in Fig. 37.19, and is a good model for the soft contact described in Sect. 37.6.1. More details can be found in [37.35–37, 39–42].

37.8 Contacts in Grasping and Fixture Designs

It is often important to relate the force and contact displacement (or deformation of contact interface) in grasping and fixture design in which deformable contacts are concerned. Furthermore, such a force–displacement relationship is typically nonlinear due to the nature of contact. A linear expression such as Hooke's law cannot capture the instantaneous and overall characteristics of force and displacement for these contacts. In this section, we formulate and discuss such a relationship using the elastic contact model.

From Fig. 37.13 and the geometry of contact, (37.17), relating the contact radius with normal force, can be rewritten. Assuming $d \ll R_0$, the following equation can be derived [37.39]

$$N = c_d\, d^{\zeta}\,, \tag{37.37}$$

where c_d is a proportional constant, and ζ is

$$\zeta = \frac{1}{2\gamma}\,. \tag{37.38}$$

Both c_d and ζ can be obtained experimentally. The exponent ζ can also be obtained from γ through (37.38)

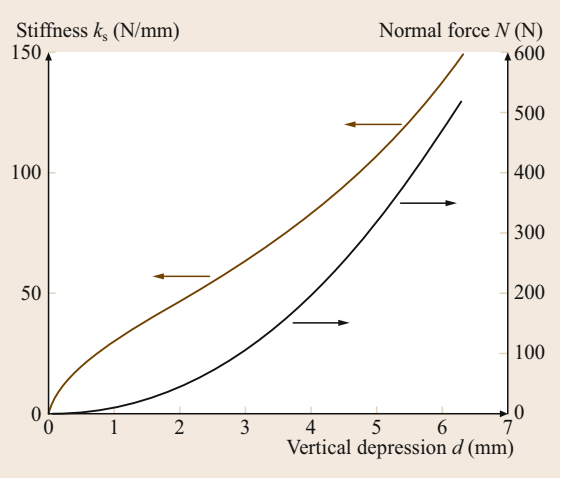


Fig. 37.20 The typical load–depression curve in (37.37) shown with a *scale to the right*, and the contact stiffness, k_s, as given by (37.39) with a *scale to the left*

if γ is already known for the fingertip. The range of the exponent in (37.37) is $3/2 \le \zeta < \infty$. In (37.37), the approach or the vertical depression of the fingertip, d, is proportional to the normal force raised to the power of 2γ (cf. (37.38)) which ranges from 0 to 2/3.

A plot of normal force versus displacement of contact is shown in Fig. 37.20 using (37.37) with a nonlinear power-law equation. Experimental data of contact between a soft finger and flat surface show consistent results when compared with Fig. 37.20 [37.39].

37.8.1 Contact Stiffness of Soft Fingers

The nonlinear contact stiffness of a soft finger is defined as the ratio of the change in normal force with respect to the change in vertical depression at the contact. The contact stiffness of soft fingers can be obtained by differentiating (37.37) as follows

$$k_s = \frac{\partial N}{\partial d} = c_d\, \zeta\, d^{\zeta-1} = \zeta \left(\frac{N}{d}\right). \tag{37.39}$$

Substitute (37.37) with $d = \left(\frac{N}{c_d}\right)^{1/\zeta}$ into (37.39) to derive

$$k_s = c_d^{\frac{1}{\zeta}}\, \zeta\, N^{\frac{\zeta-1}{\zeta}} = c_d^{2\gamma}\, \zeta\, N^{1-2\gamma}\,. \tag{37.40}$$

Thus, the nonlinear contact stiffness of soft fingers, derived in a succinct form in (37.39), is the product of the exponent ζ and the ratio of the normal force versus the approach, N/d. A typical contact stiffness as a function of vertical depression is plotted in Fig. 37.20. The stiffness shown in the figure increases with the force. The expressions for the stiffness, k_s, as a function of various parameter(s) are summarized in the following Table 37.3.

Table 37.3 Summary of equations of contact stiffness

	$f(d)$	$f(N)$	$f(N,d)$
k_s	$c_d \zeta d^{\zeta-1}$	$c_d^{\frac{1}{\zeta}}\, \zeta N^{\frac{\zeta-1}{\zeta}}$	$\zeta\left(\frac{N}{d}\right)$

Avidbots Ex1016
Page 981 of 2154

Table 37.4 Summary of the contact mechanics equations for linear elastic (when $\gamma = 1/3$ or $\zeta = 3/2$) and nonlinear elastic soft fingers

Description	Equation for soft fingers	Parameters
Power law	$a = c\,N^{\gamma}$	$0 \le \gamma \le \frac{1}{3}$
Pressure distribution	$p(r) = p(0)\left[1 - \left(\frac{r}{a}\right)^{k}\right]^{\frac{1}{k}}$	Typically $k \ge 1.8$
Contact approach	$N = c_{\mathrm{d}}\, d^{\zeta}$	$\frac{3}{2} \le \zeta \le \infty$
Contact stiffness	$k_{\mathrm{s}} = \zeta \frac{N}{d}$	Nonlinear

The change in stiffness can be obtained by differentiating (37.39) to derive the following equations

$$\frac{\partial k_{\mathrm{s}}}{\partial d} = c_{\mathrm{d}}\,\zeta(\zeta - 1)\,d^{\zeta-2}\,, \tag{37.41}$$

$$\frac{\partial k_{\mathrm{s}}}{\partial N} = \frac{\zeta - 1}{d}\,. \tag{37.42}$$

Equation (37.39) suggests that the stiffness of soft contacts always increases because the ratio N/d always increases, with constant ζ for prescribed fingertip material and range of normal force. This is consistent with the observation that the contact stiffness becomes larger (i. e., stiffer) with larger depression and force. Additionally, (37.42) suggests that the contact stiffness k_{s} always increases with the normal force because $\partial k_{\mathrm{s}}/\partial N > 0$ owing to $\zeta \ge 1.5$. Moreover, the change in stiffness with respect to the normal load is inversely proportional to the vertical depression, d, as derived in (37.42). This result asserts that the rate of increase in the contact stiffness will gradually become less and less as the normal load and vertical depression increase.

A summary of the equations for soft fingers is presented in Table 37.4. The range of the exponent of the general power-law equation in (37.17) is $0 \le \gamma \le 1/3$. The Hertzian contact has an exponent of $\gamma = 1/3$; therefore, the Hertzian contact theory for linear elastic materials is a special case of (37.17). Linear elastic materials, such as steel or other metallic fingertips, under small deformation, generally follow the Hertzian contact theory fairly well. The power-law equation in (37.17) should be employed for fingertips made of softer materials, such as soft rubber or silicone or even viscoelastic fingertips.

37.8.2 Application of Soft Contact Theory to Fixture Design

The preceding analysis and results can be applied in fixturing design and other applications that involve contact with finite areas [37.6]. In fixture design with soft contacts (e.g., copper surfaces) under relatively large deformation and load, the power-law equations in Table 37.4 should be considered in place of the Hertzian contact equation. In these cases, the Hertzian contact model is no longer accurate and should be replaced. On the other hand, if linear elastic materials are used in fixture design with relatively small deformation ($d/R_0 \le 5\%$), the exponent should be taken as $\gamma = 1/3$ when applying the contact theory. Furthermore, the exponent γ was found to be material dependent and not geometry dependent in general [37.38]. Once the value of γ is determined for the material (for example, using a tensile testing machine with the experimental procedure in [37.38]), it can be employed for the analysis of fixturing design using the relevant equations presented herewith.

37.9 Conclusions and Further Reading

In this chapter, both rigid-body and elastic models of contact interfaces are discussed, including kinematic constraints and the duality between the contact wrenches and twists. Contact forces that may arise with Coulomb friction are described. Multiple-contact manipulation tasks with rigid-body and Coulomb friction are presented. Soft and viscoelastic contact interfaces are presented and discussed. The friction limit surface is introduced and utilized to analyze pushing problems. An example of constructing a friction limit surface for soft contact is presented based on the formulation of force/moment of soft contact interface. Applications of these contact models in fixture analysis and design are presented with the modeling of contact interface. Many references in the bibliography section of this chapter provide further reading on this subject.

Manipulation modes include grasping, pushing, rolling, batting, throwing, catching, casting, and other kinds of quasistatic and dynamic manipulation [37.110]. This chapter presents an overview of contact modeling interfaces, with a particular focus on their use in manipulation tasks, including graspless or *nonprehensile* manipulation modes such as pushing. Analyses of grasps and fixtures are also presented. *Mason*'s textbook [37.14] expands on a number of themes in this chapter, including the theory of polyhedral convex cones, graphical methods for planar problems, and

applications to manipulation planning. Fundamental material on polyhedral cones can be found in [37.111] and their application to representations of twists and wrenches in [37.10, 112–114]. Twists and wrenches are elements of classical *screw theory*, which is covered in the texts [37.115, 116] and from the point of view of robotics in the texts [37.30, 117, 118].

In addition, several relevant chapters in this handbook provide further reading for theory and applications, for example, Chaps. 2 and 38.

Video-References

VIDEO 802 Pushing, sliding, and toppling
available from http://handbookofrobotics.org/view-chapter/37/videodetails/802

VIDEO 803 Horizontal Transport by 2-DOF Vibration
available from http://handbookofrobotics.org/view-chapter/37/videodetails/803

VIDEO 804 Programmable Velocity Vector Fields by 6-DOF Vibration
available from http://handbookofrobotics.org/view-chapter/37/videodetails/804

References

37.1 A. Bicchi: On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation, Int. J. Robotics Auton. Syst. **13**, 127–147 (1994)

37.2 K. Harada, M. Kaneko, T. Tsuji: Rolling based manipulation for multiple objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 3888–3895

37.3 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robotic hand, IEEE Trans. Robotics Autom. **5**(2), 151–165 (1989)

37.4 M.R. Cutkosky, S.-H. Lee: Fixture planning with friction for concurrent product/process design, Proc. NSF Eng. Des. Res. Conf. (1989)

37.5 S.-H. Lee, M. Cutkosky: Fixture planning with friction, ASME J. Eng. Ind. **113**(3), 320–327 (1991)

37.6 Q. Lin, J.W. Burdick, E. Rimon: A stiffness-based quality measure for compliant grasps and fixtures, IEEE Trans. Robotics Autom. **16**(6), 675–688 (2000)

37.7 P. Lötstedt: Coulomb friction in two-dimensional rigid body systems, Z. Angew. Math. Mech. **61**, 605–615 (1981)

37.8 P. Lötstedt: Mechanical systems of rigid bodies subject to unilateral constraints, SIAM J. Appl. Math. **42**(2), 281–296 (1982)

37.9 P.E. Dupont: The effect of Coulomb friction on the existence and uniqueness of the forward dynamics problem, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 1442–1447

37.10 M.A. Erdmann: On a representation of friction in configuration space, Int. J. Robotics Res. **13**(3), 240–271 (1994)

37.11 K.M. Lynch, M.T. Mason: Pulling by pushing, slip with infinite friction, and perfectly rough surfaces, Int. J. Robotics Res. **14**(2), 174–183 (1995)

37.12 J.S. Pang, J.C. Trinkle: Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with Coulomb friction, Math. Prog. **73**, 199–226 (1996)

37.13 J.C. Trinkle, J.S. Pang, S. Sudarsky, G. Lo: On dynamic multi-rigid-body contact problems with Coulomb friction, Z. Angew. Math. Mech. **77**(4), 267–279 (1997)

37.14 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambrige 2001)

37.15 Y.-T. Wang, V. Kumar, J. Abel: Dynamics of rigid bodies undergoing multiple frictional contacts, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 2764–2769

37.16 T.H. Speeter: Three-dimensional finite element analysis of elastic continua for tactile sensing, Int. J. Robotics Res. **11**(1), 1–19 (1992)

37.17 K. Dandekar, A.K. Srinivasan: A 3-dimensional finite element model of the monkey fingertip for predicting responses of slowly adapting mechanoreceptors, ASME Bioeng. Conf., Vol. 29 (1995) pp. 257–258

37.18 N. Xydas, M. Bhagavat, I. Kao: Study of soft-finger contact mechanics using finite element analysis and experiments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000)

37.19 K. Komvopoulos, D.-H. Choi: Elastic finite element analysis of multi-asperity contacts, J. Tribol. **114**, 823–831 (1992)

37.20 L.T. Tenek, J. Argyris: *Finite Element Analysis for Composite Structures* (Kluwer, Bosten 1998)

37.21 Y. Nakamura: Contact stability measure and optimal finger force control of multi-fingered robot hands, crossing bridges: Advances in flexible automation and robotics, Proc. U.S.-Jpn. Symp. Flex. Autom. (1988) pp. 523–528

37.22 Y.C. Park, G.P. Starr: Optimal grasping using a multifingered robot hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 689–694

37.23 E. Rimon, J. Burdick: On force and form closure for multiple finger grasps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 1795–1800

37.24 E. Rimon, J.W. Burdick: New bounds on the number of frictionless fingers required to immobilize

Avidbots Ex1016
Page 983 of 2154

planar objects, J. Robotics Sys. **12**(6), 433–451 (1995)
37.25 E. Rimon, J.W. Burdick: Mobility of bodies in contact – Part I: A 2nd-order mobility index for multiple-finger grasps, IEEE Trans. Robotics Autom. **14**(5), 696–708 (1998)
37.26 D.J. Montana: The kinematics of contact and grasp, Int. J. Robotics Res. **7**(3), 17–32 (1988)
37.27 C.S. Cai, B. Roth: On the planar motion of rigid bodies with point contact, Mech. Mach. Theory **21**(6), 453–466 (1986)
37.28 C. Cai, B. Roth: On the spatial motion of a rigid body with point contact, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 686–695
37.29 A.B.A. Cole, J.E. Hauser, S.S. Sastry: Kinematics and control of multifingered hands with rolling contact, IEEE Trans. Autom. Control **34**(4), 398–404 (1989)
37.30 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)
37.31 F. Reuleaux: *The Kinematics of Machinery* (Dover, New York 1963), reprint of MacMillan, 1876
37.32 C.A. Coulomb: *Theorie des Machines Simples en Ayant Egard au Frottement de Leurs Parties et a la Roideur des Cordages* (Bachelier, Paris 1821)
37.33 Y. Maeda, T. Arai: Planning of graspless manipulation by a multifingered robot hand, Adv. Robotics **19**(5), 501–521 (2005)
37.34 M.T. Mason: Two graphical methods for planar contact problems, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Osaka (1991) pp. 443–448
37.35 R. Howe, I. Kao, M. Cutkosky: Sliding of robot fingers under combined torsion and shear loading, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1, Philadelphia (1988) pp. 103–105
37.36 I. Kao, M.R. Cutkosky: Dextrous manipulation with compliance and sliding, Int. J. Robotics Res. **11**(1), 20–40 (1992)
37.37 R.D. Howe, M.R. Cutkosky: Practical force-motion models for sliding manipulation, Int. J. Robotics Res. **15**(6), 555–572 (1996)
37.38 N. Xydas, I. Kao: Modeling of contact mechanics and friction limit surface for soft fingers with experimental results, Int. J. Robotics Res. **18**(9), 941–950 (1999)
37.39 I. Kao, F. Yang: Stiffness and contact mechanics for soft fingers in grasping and manipulation, IEEE Trans. Robotics Autom. **20**(1), 132–135 (2004)
37.40 J. Jameson, L. Leifer: Quasi-Static Analysis: A method for predicting grasp stability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 876–883
37.41 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction: Part 2, Dynamics of motion Wear **143**, 331–352 (1991)
37.42 P. Tiezzi, I. Kao: Modeling of viscoelastic contacts and evolution of limit surface for robotic contact interface, IEEE Trans. Robotics **23**(2), 206–217 (2007)
37.43 M. Anitescu, F. Potra: Formulating multi-rigid-body contact problems with friction as solvable linear complementarity problems, ASME J. Nonlin. Dyn. **14**, 231–247 (1997)
37.44 S. Berard, J. Trinkle, B. Nguyen, B. Roghani, J. Fink, V. Kumar: daVinci code: A multi-model simulation and analysis tool for multi-body systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)
37.45 P. Song, J.-S. Pang, V. Kumar: A semi-implicit time-stepping model for frictional compliant contact problems, Int. J. Numer. Methods Eng. **60**(13), 2231–2261 (2004)
37.46 D. Stewart, J. Trinkle: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and Coulomb friction, Int. J. Numer. Methods Eng. **39**, 2673–2691 (1996)
37.47 R.W. Cottle, J.-S. Pang, R.E. Stone: *The Linear Complementarity Problem* (Academic, New York 1992)
37.48 S.N. Simunovic: Force information in assembly processes, Int. Symp. Ind. Robots (1975)
37.49 V.-D. Nguyen: Constructing force-closure grasps, Int. J. Robotics Res. **7**(3), 3–16 (1988)
37.50 K.M. Lynch: Toppling manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999)
37.51 M.T. Zhang, K. Goldberg, G. Smith, R.-P. Berretty, M. Overmars: Pin design for part feeding, Robotica **19**(6), 695–702 (2001)
37.52 D. Reznik, J. Canny: The Coulomb pump: A novel parts feeding method using a horizontally-vibrating surface, Proc. IEEE Int. Conf. Robotics Autom. (1998) pp. 869–874
37.53 A.E. Quaid: A miniature mobile parts feeder: Operating principles and simulation results, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 2221–2226
37.54 D. Reznik, J. Canny: A flat rigid plate is a universal planar manipulator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 1471–1477
37.55 D. Reznik, J. Canny: C'mon part, do the local motion!, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 2235–2242
37.56 T. Vose, P. Umbanhowar, K.M. Lynch: Vibration-induced frictional force fields on a rigid plate, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)
37.57 M.T. Mason: Mechanics and planning of manipulator pushing operations, Int. J. Robotics Res. **5**(3), 53–71 (1986)
37.58 K.Y. Goldberg: Orienting polygonal parts without sensors, Algorithmica **10**, 201–225 (1993)
37.59 R.C. Brost: Automatic grasp planning in the presence of uncertainty, Int. J. Robotics Res. **7**(1), 3–17 (1988)
37.60 J.C. Alexander, J.H. Maddocks: Bounds on the friction-dominated motion of a pushed object, Int. J. Robotics Res. **12**(3), 231–248 (1993)
37.61 M.A. Peshkin, A.C. Sanderson: The motion of a pushed, sliding workpiece,, IEEE J. Robotics Autom. **4**(6), 569–598 (1988)
37.62 M.A. Peshkin, A.C. Sanderson: Planning robotic manipulation strategies for workpieces that slide, IEEE J. Robotics Autom. **4**(5), 524–531 (1988)

37.63 M. Brokowski, M. Peshkin, K. Goldberg: Curved fences for part alignment, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 467–473

37.64 K.M. Lynch: The mechanics of fine manipulation by pushing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 2269–2276

37.65 K.M. Lynch, M.T. Mason: Stable pushing: Mechanics, controllability, and planning, Int. J. Robotics Res. **15**(6), 533–556 (1996)

37.66 K. Harada, J. Nishiyama, Y. Murakami, M. Kaneko: Pushing multiple objects using equivalent friction center, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2485–2491

37.67 J.D. Bernheisel, K.M. Lynch: Stable transport of assemblies: Pushing stacked parts, IEEE Trans. Autom. Sci. Eng. **1**(2), 163–168 (2004)

37.68 J.D. Bernheisel, K.M. Lynch: Stable transport of assemblies by pushing, IEEE Trans. Robotics **22**(4), 740–750 (2006)

37.69 H. Mayeda, Y. Wakatsuki: Strategies for pushing a 3D block along a wall, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Osaka (1991) pp. 461–466

37.70 H. Hertz: On the Contact of Rigid Elastic Solids and on Hardness. In: *Miscellaneous Papers*, ed. by H. Hertz (MacMillan, London 1882) pp. 146–183

37.71 K.L. Johnson: *Contact Mechanics* (Cambridge Univ. Press, Cambridge 1985)

37.72 S.P. Timoshenko, J.N. Goodier: *Theory of Elasticity*, 3rd edn. (McGraw-Hill, New York 1970)

37.73 M.A. Meyers, K.K. Chawla: *Mechanical Behavior of Materials* (Prentice Hall, Upper Saddle River, 1999)

37.74 C.D. Tsai: Nonlinear Modeling on Viscoelastic Contact Interface: Theoretical Study and Experimental Validation, Ph.D. Thesis (Stony Brook University, Stony Brook 2010)

37.75 C. Tsai, I. Kao, M. Higashimori, M. Kaneko: Modeling, sensing and interpretation of viscoelastic contact interface, J. Adv. Robotics **26**(11/12), 1393–1418 (2012)

37.76 Y.C. Fung: *Biomechanics: Mechanical Properties of Living Tissues* (Springer, Berlin, Heidelberg 1993)

37.77 W. Flugge: *Viscoelasticity* (Blaisdell, Waltham 1967)

37.78 J.C. Maxwell: On the dynamical theory of gases, Philos. Trans. R. Soc. Lond. **157**, 49–88 (1867)

37.79 N. Sakamoto, M. Higashimori, T. Tsuji, M. Kaneko: An optimum design of robotic hand for handling a visco-elastic object based on maxwell model, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 1219–1225

37.80 D.P. Noonan, H. Liu, Y.H. Zweiri, K.A. Althoefer, L.D. Seneviratne: A dual-function wheeled probe for tissue viscoelastic property indentification during minimally invasive surgery, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2629–2634

37.81 P. Tiezzi, I. Kao: Characteristics of contact and limit surface for viscoelastic fingers, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006) pp. 1365–1370

37.82 P. Tiezzi, I. Kao, G. Vassura: Effect of layer compliance on frictional behavior of soft robotic fingers, Adv. Robotics **21**(14), 1653–1670 (2007)

37.83 M. Kimura, Y. Sugiyama, S. Tomokuni, S. Hirai: Constructing rheologically deformable virtual objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3737–3743

37.84 W.N. Findley, J.S.Y. Lay: A modified superposition principle applied to creep of non-linear viscoelastic material under abrupt changes in state of combined stress, Trans. Soc. Rheol. **11**(3), 361–380 (1967)

37.85 D.B. Adolf, R.S. Chambers, J. Flemming: Potential energy clock model: Justification and challenging predictions, J. Rheol. **51**(3), 517–540 (2007)

37.86 A.Z. Golik, Y.F. Zabashta: A molecular model of creep and stress relaxation in crystalline polymers, Polym. Mech. **7**(6), 864–869 (1971)

37.87 B.H. Zimm: Dynamics of polymer molecules in dilute solution: Viscoelasticity, flow birefringence and dielectric loss, J. Chem. Phys. **24**(2), 269–278 (1956)

37.88 T. Alfrey: A molecular theory of the viscoelastic behavior of an amorphous linear polymer, J. Chem. Phys. **12**(9), 374–379 (1944)

37.89 P.E. Rouse Jr.: A theory of the linear viscoelastic properties of dilute solutions of coiling polymers, J. Chem. Phys. **21**(7), 1272–1280 (1953)

37.90 F. Bueche: The viscoelastic properties of plastics, J. Chem. Phys. **22**(4), 603–609 (1954)

37.91 L.R.G. Treloar: *The Physics of Rubber Elasticity* (Clarendon Press, Oxford, 1975)

37.92 T.G. Goktekin, A.W. Bargteil, J.F. O'Brien: A method for animating viscoelastic fluid, ACM Trans. Graph. **23**(3), 463–468 (1977)

37.93 S. Arimoto, P.A.N. Nguyen, H.Y. Han, Z. Doulgeri: Dynamics and control of a set of dual fingers with soft tips, Robotica **18**, 71–80 (2000)

37.94 T. Inoue, S. Hirai: Modeling of soft fingertip for object manipulation using tactile sensig, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas, Nevada (2003)

37.95 T. Inoue, S. Hirai: Rotational contact model of soft fingertip for tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 2957–2962

37.96 T. Inoue, S. Hirai: Elastic model of deformable fingertip for soft-fingered manipulation, IEEE Trans. Robotics **22**, 1273–1279 (2006)

37.97 T. Inoue, S. Hirai: Dynamic stable manipulation via soft-fingered hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 586–591

37.98 V.A. Ho, D.V. Dat, S. Sugiyama, S. Hirai: Development and analysis of a sliding tactile soft fingertip embedded with a microforce/moment sensor, IEEE Trans. Robotics **27**(3), 411–424 (2011)

37.99 D. Turhan, Y. Mengi: Propagation of initially plane waves in nonhomogeneous viscoelastic media, Int. J. Solids Struct. **13**(2), 79–92 (1977)

37.100 P. Stucky, W. Lord: Finite element modeling of transient ultrasonic waves in linear viscoelastic media, IEEE Trans. Ultrason. Ferroelectr. Freq. Control **48**(1), 6–16 (2001)

Avidbots Ex1016
Page 985 of 2154

37.101 J.M. Pereira, J.J. Mansour, B.R. Davis: Dynamic measurement of the viscoelastic properties of skin, J. Biomech. **24**(2), 157–162 (1991)

37.102 R. Fowles, R.F. Williams: Plane stress wave propagation in solids, J. Appl. Phys. **41**(1), 360–363 (1970)

37.103 E. Wolf: *Progress in Optics* (North-Holland, Amsterdam 1992)

37.104 E.J. Nicolson, R.S. Fearing: The reliability of curvature estimates from linear elastic tactile sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995)

37.105 M. Abramowitz, I. Stegun: *Handbook of Mathematical Functions with Formulas, Graphs, and mathematical Tables*, 7th edn. (Dover, New York 1972)

37.106 I. Kao, S.-F. Chen, Y. Li, G. Wang: Application of bio-engineering contact interface and MEMS in robotic and human augmented systems, IEEE Robotics Autom, Mag. **10**(1), 47–53 (2003)

37.107 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction: Part 1. Limit surface and moment function, Wear **143**, 307–330 (1991)

37.108 J.W. Jameson: Analytic Techniques for Automated Grasp. Ph.D. Thesis (Department of Mechanical Engineering, Stanford University, Stanford 1985)

37.109 S. Goyal, A. Ruina, J. Papadopoulos: Limit surface and moment function description of planar sliding, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 794–799

37.110 K.M. Lynch, M.T. Mason: Dynamic nonprehensile manipulation: Controllability, planning, and experiments, Int. J. Robotics Res. **18**(1), 64–92 (1999)

37.111 A.J. Goldman, A.W. Tucker: Polyhedral convex cones. In: *Linear Inequalities and Related Systems*, ed. by H.W. Kuhn, A.W. Tucker (Princeton Univ. Press, Princeton 1956)

37.112 M.A. Erdman: A configuration space friction cone, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Osaka (1991) pp. 455–460

37.113 M.A. Erdmann: Multiple-point contact with friction: Computing forces and motions in configuration space, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Yokohama (1993) pp. 163–170

37.114 S. Hirai, H. Asada: Kinematics and statics of manipulation using the theory of polyhedral convex cones, Int. J. Robotics Res. **12**(5), 434–447 (1993)

37.115 R.S. Ball: *The Theory of Screws* (Cambridge Univ. Press, Cambridge 1900)

37.116 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Oxford Univ. Press, Oxford 1978)

37.117 J.K. Davidson, K.H. Hunt: *Robots and Screw Theory* (Oxford Univ. Press, Oxford 2004)

37.118 J.M. Selig: *Geometric Fundamentals of Robotics*, 2nd edn. (Springer, Berlin, Heidelberg 2005)

Avidbots Ex1016
Page 986 of 2154

Multimedia Contents

38. Grasping

Domenico Prattichizzo, Jeffrey C. Trinkle

This chapter introduces fundamental models of grasp analysis. The overall model is a coupling of models that define contact behavior with widely used models of rigid-body kinematics and dynamics. The contact model essentially boils down to the selection of components of contact force and moment that are transmitted through each contact. Mathematical properties of the complete model naturally give rise to five primary grasp types whose physical interpretations provide insight for grasp and manipulation planning.

After introducing the basic models and types of grasps, this chapter focuses on the most important grasp characteristic: complete restraint. A grasp with complete restraint prevents loss of contact and thus is very secure. Two primary restraint properties are *form closure* and *force closure*. A form closure grasp guarantees maintenance of contact as long as the links of the hand and the object are well-approximated as rigid and as long as the joint actuators are sufficiently strong. As will be seen, the primary difference between form closure and force closure grasps is the latter's reliance on contact friction. This translates into requiring fewer contacts to achieve force closure than form closure.

The goal of this chapter is to give a thorough understanding of the all-important grasp properties of form and force closure. This will be done through detailed derivations of grasp models and discussions of illustrative examples. For an in-depth historical perspective and a treasure-trove bibliography of papers addressing a wide range of topics in grasping, the reader is referred to [38.1].

38.1 **Models and Definitions** 956
38.1.1 Velocity Kinematics 957
38.1.2 Dynamics and Equilibrium 960
38.2 **Controllable Twists and Wrenches** 961
38.2.1 Grasp Classifications 962
38.2.2 Limitations of Rigid-Body Formulation 963
38.2.3 Desirable Properties 963
38.3 **Compliant Grasps** 965
38.4 **Restraint Analysis** 967
38.4.1 Form Closure 968
38.4.2 Force Closure 972
38.5 **Examples** 975
38.5.1 Example 1: Grasped Sphere 975
38.5.2 Example 2: Grasped Polygon in the Plane 978
38.5.3 Example 3: Hyperstatic Grasps 980
38.5.4 Example 4: Duality 983
38.5.5 Example 5: Form Closure 983
38.6 **Conclusion and Further Reading** 985
Video-References 986
References 986

Mechanical hands were developed to give robots the ability to grasp objects of varying geometric and physical properties. The first robotic hand designed for dexterous manipulation was the Salisbury Hand (Fig. 38.1) [38.2]. It has three three-jointed fingers; enough to control all six degrees of freedom of an object and the grip pressure. The fundamental grasp modeling and analysis done by Salisbury provides a basis for grasp synthesis and dexterous manipulation research which continues today. Some of the most ma-

Avidbots Ex1016
Page 987 of 2154

ture analysis techniques are embedded in the widely used software *GraspIt!* [38.3] and *SynGrasp* [38.4, 5]. *GraspIt!* contains models for several robot hands and provides tools for grasp selection, dynamic grasp simulation, and visualization. SynGrasp is a MATLAB Toolbox that can be obtained from [38.6] and provides models and functions for grasp analysis with both fully and underactuated hands. It can be a useful educational tool to get aquainted with the mathematical framework of robotic grasping described in this Chapter. Over the years since the Salisbury Hand was built, many articulated robot hands have been developed. Nearly all of these have one actuator per joint or fewer. A notable exception is the *DLR hand arm system* developed by DLR (Deutsches Zentrum für Luft- und Raumfahrt), which has two actuators per joint that drive each joint independently with two antagonistic tendons (Fig. 38.7) [38.7].

38.1 Models and Definitions

A mathematical model of grasping must be capable of predicting the behavior of the hand and object under various loading conditions that may arise during grasping. Generally, the most desirable behavior is grasp maintenance in the face of unknown disturbing forces and moments applied to the object. Typically these disturbances arise from inertia forces which become appreciable during high-speed manipulation or applied forces such as those due to gravity. Grasp maintenance means that the contact forces applied by the hand are such that they prevent contact separation and unwanted contact sliding. The special class of grasps that can be maintained for every possible disturbing load is known as closure grasps. Figure 38.1 shows the Salisbury Hand [38.2, 8], executing a closure grasp of an object by wrapping its fingers around it and pressing it against its palm. Formal definitions, analysis, and computational tests for closure will be presented in Sect. 38.4.

Figure 38.2 illustrates some of the main quantities that will be used to model grasping systems. Assume that the links of the hand and the object are rigid and that there is a unique, well-defined tangent plane at each contact point. Let $\{\boldsymbol{N}\}$ represent a conveniently chosen inertial frame fixed in the workspace. The frame $\{\boldsymbol{B}\}$ is fixed to the object with its origin defined relative to $\{\boldsymbol{N}\}$ by the vector $\boldsymbol{p} \in \mathbb{R}^3$, where $\mathbb{R}^3$ denotes three-dimensional Euclidean space. A convenient choice for $\boldsymbol{p}$ is the center of mass of the object. The position of contact point i in $\{\boldsymbol{N}\}$ is defined by the vector $\boldsymbol{c}_i \in \mathbb{R}^3$. At contact point i, we define a frame $\{\boldsymbol{C}\}_i$, with axes $\{\hat{\boldsymbol{n}}_i, \hat{\boldsymbol{t}}_i, \hat{\boldsymbol{o}}_i\}$ ($\{\boldsymbol{C}\}_1$ is shown in *exploded* view). The unit vector $\hat{\boldsymbol{n}}_i$ is normal to the contact tangent plane and directed toward the object. The other two unit vectors are orthogonal and lie in the tangent plane of the contact.

Let the joints be numbered from 1 to n_q. Denote by $\boldsymbol{q} = [q_1 \cdots q_{n_q}]^{\mathrm{T}} \in \mathbb{R}^{n_q}$ the vector of joint displacements, where the superscript ^{T} indicates matrix transposition. Also, let $\boldsymbol{\tau} = [\tau_1 \cdots \tau_{n_q}]^{\mathrm{T}} \in \mathbb{R}^{n_q}$ represent joint loads (forces in prismatic joints and torques in revolute joints). These loads can result from actuator actions, other applied forces, and inertia forces. They could also arise from interaction at the contacts between the object and hand. However, it will be convenient to sep-

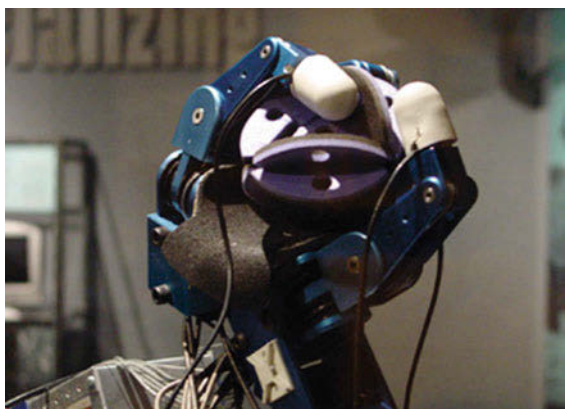


Fig. 38.1 The Salisbury Hand grasping an object

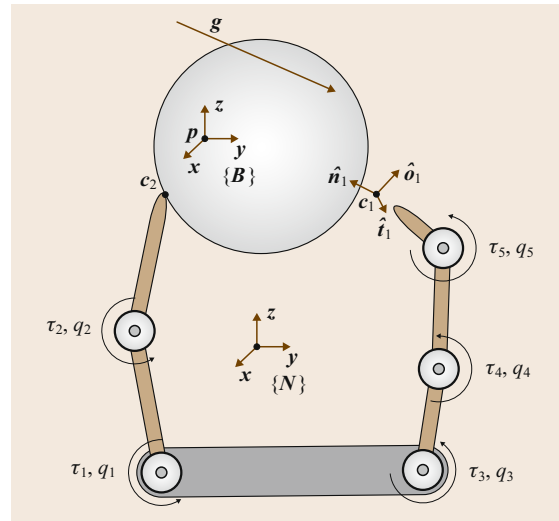


Fig. 38.2 Main quantities for grasp analysis

Avidbots Ex1016
Page 988 of 2154

arate joint loads into two components: those arising from contacts and those arising from all other sources. Throughout this Chapter, non-contact loads will be denoted by $\boldsymbol{\tau}$.

Let $\boldsymbol{u} \in \mathbb{R}^{n_u}$ denote the vector describing the position and orientation of $\{\boldsymbol{B}\}$ relative to $\{\boldsymbol{N}\}$. For planar systems $n_u = 3$. For spatial systems, n_u is three plus the number of parameters used to represent orientation; typically three (for Euler angles) or four (for unit quaternions). Denote by $\boldsymbol{v} = [\boldsymbol{v}^{\mathrm{T}}\ \boldsymbol{\omega}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{n_v}$, the twist of the object described in $\{\boldsymbol{N}\}$. It is composed of the translational velocity $\boldsymbol{v} \in \mathbb{R}^3$ of the point $\boldsymbol{p}$ and the angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ of the object, both expressed in $\{\boldsymbol{N}\}$. A twist of a rigid body can be referred to any convenient frame fixed to the body. The components of the referred twist represent the velocity of the origin of the new frame and the angular velocity of the body, both expressed in the new frame (Table 38.1). For a rigorous treatment of twists and wrenches see [38.9, 10]. Note that for planar systems, $\boldsymbol{v} \in \mathbb{R}^2$, $\omega \in \mathbb{R}$, and so $n_v = 3$.

Another important point is $\dot{\boldsymbol{u}} \neq \boldsymbol{v}$. Instead, these variables are related by the matrix $\mathbf{V}$ as

$$\dot{\boldsymbol{u}} = \mathbf{V}\boldsymbol{v}\,, \tag{38.1}$$

where the matrix $\mathbf{V} \in \mathbb{R}^{n_u \times n_v}$ is not generally square, but nonetheless satisfies $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}$ [38.11], where $\mathbf{I}$ is the identity matrix, and the dot over the $\boldsymbol{u}$ implies differentiation with respect to time. Note that for planar systems, $\mathbf{V} = \mathbf{I} \in \mathbb{R}^{3\times 3}$.

Let $\boldsymbol{f} \in \mathbb{R}^3$ be the force applied to the object at the point $\boldsymbol{p}$ and let $\boldsymbol{m} \in \mathbb{R}^3$ be the applied moment. These are combined into the object load, or wrench, vector denoted by $\boldsymbol{g} = \left[\boldsymbol{f}^{\mathrm{T}}\ \boldsymbol{m}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{n_v}$, where $\boldsymbol{f}$ and $\boldsymbol{m}$ are expressed in $\{\boldsymbol{N}\}$. Like twists, wrenches can be referred to any convenient frame fixed to the body. One can think of this as translating the line of application of the force until it contains the origin of the new frame, then adjusting the moment component of the wrench to offset the moment induced by moving the line of the force. Last, the force and adjusted moment are expressed in the new frame. As done with the joint loads, the object wrench will be partitioned into two main parts: contact and non-contact wrenches. Throughout this chapter, $\boldsymbol{g}$ will denote the non-contact wrench on the object.

38.1.1 Velocity Kinematics

The material in this chapter is valid for a wide range of robot hands and other grasping mechanisms. The hand is assumed to be composed of a palm that serves as the common base for any number of fingers, each with any number of joints. The formulations given in this chapter are expressed explicitly in terms of only revolute and prismatic joints. However, most other common joints can be modeled by combinations of revolute and prismatic joints (e.g., cylindrical and planar). Any number of contacts may occur between any link and the object.

Table 38.1 Primary notation for grasp analysis

Notation	Definition
n_{c}	Number of contacts
n_q	Number of joints of the hand
n_v	Number of degrees of freedom of object
n_λ	Number of contact wrench components
$\boldsymbol{q} \in \mathbb{R}^{n_q}$	Joint displacements
$\dot{\boldsymbol{q}} \in \mathbb{R}^{n_q}$	Joint velocities
$\boldsymbol{\tau} \in \mathbb{R}^{n_q}$	Non-contact joint loads
$\boldsymbol{u} \in \mathbb{R}^{n_u}$	Position and orientation of object
$\boldsymbol{v} \in \mathbb{R}^{n_v}$	Twist of object
$\boldsymbol{g} \in \mathbb{R}^{n_v}$	Non-contact object wrench
$\boldsymbol{\lambda} \in \mathbb{R}^{n_\lambda}$	Transmitted contact wrenches
$\boldsymbol{v}_{\mathrm{cc}} \in \mathbb{R}^{n_\lambda}$	Transmitted contact twists
$\{\boldsymbol{B}\}$	Frame fixed in object
$\{\boldsymbol{C}\}_i$	Frame at contact i
$\{\boldsymbol{N}\}$	Inertial frame

Grasp Matrix and Hand Jacobian

Two matrices are of the utmost importance in grasp analysis: the *grasp matrix* $\mathbf{G}$ and the *hand Jacobian* $\mathbf{J}$. These matrices define the relevant velocity kinematics and force transmission properties of the contacts. The following derivations of $\mathbf{G}$ and $\mathbf{J}$ will be done under the assumption that the system is three-dimensional ($n_v = 6$). Changes for planar systems will be noted later.

Each contact should be considered as two coincident points; one on the hand and one on the object. The hand Jacobian maps the joint velocities to the twists of the hand to the contact frames, while the transpose of the grasp matrix maps the object twist to the contact frames. Finger joint motions induce a rigid body motion in each link of the hand. It is implicit in the terminology, twists of the hand, that the twist referred to contact i is the twist of the link involved in contact i. Thus these matrices can be derived from the transforms that change the reference frame of a twist.

To derive the grasp matrix, let $\boldsymbol{\omega}^{\mathrm{N}}_{\mathrm{obj}}$ denote the angular velocity of the object expressed in $\{\boldsymbol{N}\}$ and let $\boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{obj}}$, also expressed in $\{\boldsymbol{N}\}$, denote the velocity of the point on the object coincident with the origin of $\{\boldsymbol{C}\}_i$. These velocities can be obtained from the object twist referred to $\{\boldsymbol{N}\}$ as

$$\begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{obj}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{\mathrm{obj}} \end{pmatrix} = \mathbf{P}_i^{\mathrm{T}}\boldsymbol{v}, \tag{38.2}$$

Avidbots Ex1016
Page 989 of 2154

where

$$\mathbf{P}_i = \begin{pmatrix} \mathbf{I}_{3\times3} & \mathbf{0} \\ \mathbf{S}(\boldsymbol{c}_i - \boldsymbol{p}) & \mathbf{I}_{3\times3} \end{pmatrix} \tag{38.3}$$

$\mathbf{I}_{3\times3} \in \mathbb{R}^{3\times3}$ is the identity matrix, and $\mathbf{S}(\boldsymbol{c}_i - \boldsymbol{p})$ is the cross product matrix, that is, given a three-vector $\boldsymbol{r} = [r_x\, r_y\, r_z]^{\mathrm{T}}$, $\mathbf{S}(\boldsymbol{r})$ is defined as follows

$$\mathbf{S}(\boldsymbol{r}) = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix}.$$

The object twist referred to $\{\boldsymbol{C}\}_i$ is simply the vector on the left-hand side of (38.2) expressed in $\{\boldsymbol{C}\}_i$.

Let $\mathbf{R}_i = (\hat{\boldsymbol{n}}_i\, \hat{\boldsymbol{t}}_i\, \hat{\boldsymbol{o}}_i) \in \mathbb{R}^{3\times3}$ represent the orientation of the i-th contact frame $\{\boldsymbol{C}\}_i$ with respect to the inertial frame (the unit vectors $\hat{\boldsymbol{n}}_i$, $\hat{\boldsymbol{t}}_i$, and $\hat{\boldsymbol{o}}_i$ are expressed in $\{\boldsymbol{N}\}$). Then the object twist referred to $\{\boldsymbol{C}\}_i$ is given as

$$\boldsymbol{v}_{i,\mathrm{obj}} = \overline{\mathbf{R}}_i^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{obj}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{\mathrm{obj}} \end{pmatrix}, \tag{38.4}$$

where $\overline{\mathbf{R}}_i = \mathrm{Blockdiag}(\mathbf{R}_i,\ \mathbf{R}_i) = \begin{pmatrix} \mathbf{R}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_i \end{pmatrix} \in \mathbb{R}^{6\times6}$.

Substituting $\mathbf{P}_i^{\mathrm{T}}\boldsymbol{v}$ from (38.2) into (38.4) yields the partial grasp matrix $\tilde{\mathbf{G}}_i^{\mathrm{T}} \in \mathbb{R}^{6\times6}$, that maps the object twist from $\{\boldsymbol{N}\}$ to $\{\boldsymbol{C}\}_i$

$$\boldsymbol{v}_{i,\mathrm{obj}} = \tilde{\mathbf{G}}_i^{\mathrm{T}}\boldsymbol{v}\,, \tag{38.5}$$

where

$$\tilde{\mathbf{G}}_i^{\mathrm{T}} = \overline{\mathbf{R}}_i^{\mathrm{T}}\mathbf{P}_i^{\mathrm{T}}\,. \tag{38.6}$$

The hand Jacobian can be derived similarly. Let $\boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}}$ be the angular velocity of the link of the hand touching the object at contact i, expressed in $\{\boldsymbol{N}\}$, and define $\boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}}$ as the translational velocity of contact i on the hand, expressed in $\{\boldsymbol{N}\}$. These velocities are related to the joint velocities through the matrix $\mathbf{Z}_i$ whose columns are the Plücker coordinates of the axes of the joints [38.9, 10]. We have

$$\begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}} \end{pmatrix} = \mathbf{Z}_i\dot{\boldsymbol{q}}\,, \tag{38.7}$$

where $\mathbf{Z}_i \in \mathbb{R}^{6\times n_q}$ is defined as

$$\mathbf{Z}_i = \begin{pmatrix} \boldsymbol{d}_{i,1} & \cdots & \boldsymbol{d}_{i,n_q} \\ \boldsymbol{\kappa}_{i,1} & \cdots & \boldsymbol{\kappa}_{i,n_q} \end{pmatrix}, \tag{38.8}$$

with the vectors $\boldsymbol{d}_{i,j}, \boldsymbol{\kappa}_{i,j} \in \mathbb{R}^3$ defined as

$$\boldsymbol{d}_{i,j} = \begin{cases} \mathbf{0}_{3\times1} & \text{if contact force } i \text{ does not affect joint } j\,, \\ \hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is prismatic}\,, \\ \mathbf{S}(\boldsymbol{c}_i - \boldsymbol{\zeta}_j)^{\mathrm{T}}\hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is revolute}\,, \end{cases}$$

$$\boldsymbol{\kappa}_{i,j} = \begin{cases} \mathbf{0}_{3\times1} & \text{if contact force } i \text{ does not affect joint } j\,, \\ \mathbf{0}_{3\times1} & \text{if joint } j \text{ is prismatic}\,, \\ \hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is revolute}\,, \end{cases}$$

where $\boldsymbol{\zeta}_j$ is the origin of the coordinate frame associated with the j-th joint and $\hat{\boldsymbol{z}}_j$ is the unit vector in the direction of the z-axis in the same frame, as shown in Fig. 38.12. Both vectors are expressed in $\{\boldsymbol{N}\}$. These frames may be assigned by any convenient method, for example, the Denavit–Hartenberg method [38.12]. The $\hat{\boldsymbol{z}}_j$-axis is the rotational axis for revolute joints and the direction of translation for prismatic joints.

The final step in referring the hand twists to the contact frames is change the frame of expression of $\boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}}$ and $\boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}}$ to $\{\boldsymbol{C}\}_i$

$$\boldsymbol{v}_{i,\mathrm{hnd}} = \overline{\mathbf{R}}_i^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}} \end{pmatrix}. \tag{38.9}$$

Combining (38.9) and (38.7) yields the partial hand Jacobian $\tilde{\mathbf{J}}_i \in \mathbb{R}^{6\times n_q}$, which relates the joint velocities to the contact twists on the hand

$$\boldsymbol{v}_{i,\mathrm{hnd}} = \tilde{\mathbf{J}}_i\dot{\boldsymbol{q}}\,, \tag{38.10}$$

where

$$\tilde{\mathbf{J}}_i = \overline{\mathbf{R}}_i^{\mathrm{T}}\mathbf{Z}_i. \tag{38.11}$$

To simplify notation, stack all the twists on the hand and object into the vectors $\boldsymbol{v}_{c,\mathrm{hnd}} \in \mathbb{R}^{6n_{\mathrm{c}}}$ and $\boldsymbol{v}_{c,\mathrm{obj}} \in \mathbb{R}^{6n_{\mathrm{c}}}$ as follows

$$\boldsymbol{v}_{c,\xi} = \begin{pmatrix} \boldsymbol{v}_{1,\xi}^{\mathrm{T}} & \cdots & \boldsymbol{v}_{n_{\mathrm{c}},\xi}^{\mathrm{T}} \end{pmatrix}^{\mathrm{T}}, \quad \xi = \{\mathrm{obj}, \mathrm{hnd}\}\,.$$

Now the *complete grasp matrix* $\tilde{\mathbf{G}} \in \mathbb{R}^{6\times6n_{\mathrm{c}}}$ and *complete hand Jacobian* $\tilde{\mathbf{J}} \in \mathbb{R}^{6n_{\mathrm{c}}\times n_q}$ relate the various velocity quantities as follows

$$\boldsymbol{v}_{c,\mathrm{obj}} = \tilde{\mathbf{G}}^{\mathrm{T}}\boldsymbol{v}\,, \tag{38.12}$$

$$\boldsymbol{v}_{c,\mathrm{hnd}} = \tilde{\mathbf{J}}\dot{\boldsymbol{q}}\,, \tag{38.13}$$

where

$$\tilde{\mathbf{G}}^{\mathrm{T}} = \begin{pmatrix} \tilde{\mathbf{G}}_1^{\mathrm{T}} \\ \vdots \\ \tilde{\mathbf{G}}_{n_{\mathrm{c}}}^{\mathrm{T}} \end{pmatrix}, \quad \tilde{\mathbf{J}} = \begin{pmatrix} \tilde{\mathbf{J}}_1 \\ \vdots \\ \tilde{\mathbf{J}}_{n_{\mathrm{c}}} \end{pmatrix}. \tag{38.14}$$

The term *complete* is used to emphasize that all $6n_c$ twist components at the contacts are included in the mapping. See Example 1, Part 1 and Example 3, Part 1 at the end of this chapter for clarification.

Contact Modeling

Contacts play a central role in grasping. Contacts allows to impose a given motion to the object or to apply a given force through the object. All grasping actions go through contacts whose model and control is crucial in grasping. Three contact models useful for grasp analysis are reviewed here. For a complete discussion of contact modeling in robotics, readers are referred to Chap. 37.

The three models of greatest interest in grasp analysis are known as *point-contact-without-friction*, *hard-finger*, and *soft-finger* [38.13]. These models select components of the contact twists to transmit between the hand and the object. This is done by equating a subset of the components of the hand and object twist at each contact. The corresponding components of the contact force and moment are also equated, but without regard for the constraints imposed by a friction model (Sect. 38.4.2).

The point-contact-without-friction (PwoF) model is used when the contact patch is very small and the surfaces of the hand and object are slippery. With this model, only the normal component of the translational velocity of the contact point on the hand (i. e., the first component of $\boldsymbol{\nu}_{i,\mathrm{hnd}}$) is transmitted to the object. The two components of tangential velocity and the three components of angular velocity are not transmitted. Analogously, the normal component of the contact force is transmitted, but the frictional forces and moments are assumed to be negligible.

A hard finger (HF) model is used when there is significant contact friction, but the contact patch is small, so that no appreciable friction moment exists. When this model is applied to a contact, all three translational velocity components of the contact point on the hand (i. e., the first three components of $\boldsymbol{\nu}_{i,\mathrm{hnd}}$) and all three components of the contact force are transmitted through the contact. None of the angular velocity components or moment components are transmitted.

The soft finger (SF) model is used in situations in which the surface friction and the contact patch are large enough to generate significant friction forces and a friction moment about the contact normal. At a contact where this model is enforced, the three translational velocity components of the contact on the hand and the angular velocity component about the contact normal are transmitted (i. e., the first four components of $\boldsymbol{\nu}_{i,\mathrm{hnd}}$). Similarly, all three components of contact force and the normal component of the contact moment are transmitted.

Remark 38.1

The reader may see a contradiction between the rigid-body assumption and the soft-finger model. The rigid-body assumption is an approximation that simplifies all aspects of the analysis of grasping, but nonetheless is sufficiently accurate in many real situations. Without it, grasp analysis would be impractical. On the other hand, the need for a soft-finger model is a clear admission that the finger links and object are not rigid. However, it can be usefully applied in situations in which the amount of deformation required to obtain a large contact patch is small. Such situations occur when the local surface geometries are similar. If large finger or body deformations exist in the real system, the rigid-body approach presented in this chapter should be used with caution.

To develop the PwoF, HF, and SF models, define the relative twist at contact i as follows

$$\begin{pmatrix}\tilde{\mathbf{J}}_i & -\tilde{\mathbf{G}}_i^{\mathrm{T}}\end{pmatrix}\begin{pmatrix}\dot{\boldsymbol{q}}\\ \boldsymbol{\nu}\end{pmatrix} = \boldsymbol{\nu}_{i,\mathrm{hnd}} - \boldsymbol{\nu}_{i,\mathrm{obj}}\,.$$

A particular contact model is defined through the matrix $\mathbf{H}_i \in \mathbb{R}^{n_{\lambda i}\times 6}$, which selects $n_{\lambda i}$ components of the relative contact twist and sets them to zero

$$\mathbf{H}_i(\boldsymbol{\nu}_{i,\mathrm{hnd}} - \boldsymbol{\nu}_{i,\mathrm{obj}}) = \mathbf{0}\,.$$

These components are referred to as transmitted degrees of freedom (DOF). Define $\mathbf{H}_i$ as follows

$$\mathbf{H}_i = \left[\begin{array}{c|c}\mathbf{H}_{i\mathrm{F}} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{H}_{i\mathrm{M}}\end{array}\right], \tag{38.15}$$

where $\mathbf{H}_{i\mathrm{F}}$ and $\mathbf{H}_{i\mathrm{M}}$ are the translational and rotational component selection matrices. Table 38.2 gives the definitions of the selection matrices for the three contact models, where *vacuous* for $\mathbf{H}_{i\mathrm{M}}$ means that the corresponding block row matrix in (38.15) is void (i. e., it has zero rows and columns). Notice that for the SF model, $\mathbf{H}_{i\mathrm{M}}$ selects rotation about the contact normal.

After choosing a transmission model for each contact, the kinematic contact constraint equations for all n_c contacts can be written in compact form as

$$\mathbf{H}(\boldsymbol{\nu}_{c,\mathrm{hnd}} - \boldsymbol{\nu}_{c,\mathrm{obj}}) = \mathbf{0}\,, \tag{38.16}$$

Table 38.2 Selection matrices for three contact models

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	(1 0 0)	Vacuous
HF	3	$\mathbf{I}_{3\times 3}$	Vacuous
SF	4	$\mathbf{I}_{3\times 3}$	(1 0 0)

where

$$\mathbf{H} = \text{Blockdiag}(\mathbf{H}_1, \ldots, \mathbf{H}_{n_c}) \in \mathbb{R}^{n_\lambda \times 6n_c},$$

and the number of twist components n_λ transmitted through the n_c contacts is given by $n_\lambda = \sum_{i=1}^{n_c} n_{\lambda i}$.

Finally, by substituting (38.12) and (38.13) into (38.16) one gets the compact form of the velocity kinematic contact constraints

$$\begin{pmatrix}\mathbf{J} & -\mathbf{G}^{\mathrm{T}}\end{pmatrix}\begin{pmatrix}\dot{\boldsymbol{q}} \\ \boldsymbol{\nu}\end{pmatrix} = \mathbf{0}\,, \tag{38.17}$$

where the grasp matrix and hand Jacobian are finally defined as

$$\begin{aligned}\mathbf{G}^{\mathrm{T}} &= \mathbf{H}\tilde{\mathbf{G}}^{\mathrm{T}} \in \mathbb{R}^{n_\lambda \times 6}\,, \\ \mathbf{J} &= \mathbf{H}\tilde{\mathbf{J}} \in \mathbb{R}^{n_\lambda \times n_q}\,.\end{aligned} \tag{38.18}$$

For more details on the construction of $\mathbf{H}$, the grasp matrix, and the hand Jacobian, readers are referred to [38.14–16] and the references therein. Also, see Example 1, Part 2 and Example 3, Part 2.

It is worth noting that (38.17) can be written in the following form

$$\mathbf{J}\dot{\boldsymbol{q}} = \boldsymbol{\nu}_{\mathrm{cc,hnd}} = \boldsymbol{\nu}_{\mathrm{cc,obj}} = \mathbf{G}^{\mathrm{T}}\boldsymbol{\nu}\,, \tag{38.19}$$

where $\boldsymbol{\nu}_{\mathrm{cc,hnd}}$ and $\boldsymbol{\nu}_{\mathrm{cc,obj}}$ contain only the components of the twists that are transmitted by the contacts. To underline the central role of contact constraints in grasping, it is worth noting that *grasp maintenance* is defined as the situation in which constraints (38.19) are maintained over time. The kinematic contact constraint holds only if the contact force satisfies the friction constraints for the contact models with friction or the unilateral constraint for the contact model without friction (Sect. 38.4.2).

Thus, when a contact is frictionless, contact maintenance implies continued contact, but sliding is allowed. However, when a contact is of the type HF, contact maintenance implies sticking contact, since sliding would violate the HF model. Similarly, for a SF contact, there may be no sliding or relative rotation about the contact normal.

For the remainder of this chapter, it will be assumed that $\boldsymbol{\nu}_{\mathrm{cc,hnd}} = \boldsymbol{\nu}_{\mathrm{cc,obj}}$, so the notation will be shortened to $\boldsymbol{\nu}_{\mathrm{cc}}$.

Planar Simplifications

Assume that the plane of motion is the (x, y)-plane of $\{N\}$. The vectors $\boldsymbol{\nu}$ and $\boldsymbol{g}$ reduce in dimension from six to three by dropping components three, four, and five. The dimensions of vectors $\boldsymbol{c}_i$ and $\boldsymbol{p}$ reduce from three to two. The i-th rotation matrix becomes $\mathbf{R}_i = [\hat{\boldsymbol{n}}_i\, \hat{\boldsymbol{t}}_i] \in \mathbb{R}^{2\times 2}$ (where the third component of $\hat{\boldsymbol{n}}_i$ and $\hat{\boldsymbol{t}}_i$ is dropped) and (38.4) holds with $\overline{\mathbf{R}}_i = \text{Blockdiag}(\mathbf{R}_i, 1) \in \mathbb{R}^{3\times 3}$. Equation (38.2) holds with

$$\mathbf{P}_i = \begin{pmatrix}\mathbf{I}_{2\times 2} & \mathbf{0} \\ \mathbf{S}_2(\boldsymbol{c}_i - \boldsymbol{p}) & 1\end{pmatrix},$$

where $\mathbf{S}_2$ is the analog of the cross product matrix for two-dimensional vectors, given as

$$\mathbf{S}_2(\boldsymbol{r}) = \begin{pmatrix}-r_y & r_x\end{pmatrix}.$$

Equation (38.7) holds with $\boldsymbol{d}_{i,j} \in \mathbb{R}^2$ and $\boldsymbol{\kappa}_{i,j} \in \mathbb{R}$ defined as

$$\boldsymbol{d}_{i,j} = \begin{cases}\mathbf{0}_{2\times 1} & \text{if contact force } i \text{ does not affect the joint } j\,, \\ \hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is prismatic}\,, \\ \mathbf{S}(\boldsymbol{c}_i - \boldsymbol{\zeta}_j)^{\mathrm{T}} & \text{if joint } j \text{ is revolute}\,,\end{cases}$$

$$\kappa_{i,j} = \begin{cases}0 & \text{if contact force } i \text{ does not affect joint } j\,, \\ 0 & \text{if joint } j \text{ is prismatic}\,, \\ 1 & \text{if joint } j \text{ is revolute}\,.\end{cases}$$

The complete grasp matrix and hand Jacobian have reduced sizes: $\tilde{\mathbf{G}}^{\mathrm{T}} \in \mathbb{R}^{3n_c\times 3}$ and $\tilde{\mathbf{J}} \in \mathbb{R}^{3n_c\times n_q}$. As far as contact constraint is concerned, (38.15) holds with $\mathbf{H}_{i\mathrm{F}}$ and $\mathbf{H}_{i\mathrm{M}}$ defined in Table 38.3.

In the planar case, the SF and HF models are equivalent, because the object and the hand lie in a plane. Rotations about the contact normals would cause out-of-plane motions. Finally, the dimensions of the grasp matrix and hand Jacobian are reduced to the following sizes: $\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{n_\lambda\times 3}$ and $\mathbf{J} \in \mathbb{R}^{n_\lambda\times n_q}$. See Example 1, Part 3 and Example 2, Part 1.

38.1.2 Dynamics and Equilibrium

Dynamic equations of the system can be written as

$$\begin{aligned}&\mathbf{M}_{\mathrm{hnd}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{b}_{\mathrm{hnd}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda} = \boldsymbol{\tau}_{\mathrm{app}} \\ &\mathbf{M}_{\mathrm{obj}}(\boldsymbol{u})\dot{\boldsymbol{\nu}} + \boldsymbol{b}_{\mathrm{obj}}(\boldsymbol{u}, \boldsymbol{\nu}) - \mathbf{G}\boldsymbol{\lambda} = \boldsymbol{g}_{\mathrm{app}} \\ &\text{subject to constraint (38.17)}\,,\end{aligned} \tag{38.20}$$

Table 38.3 Definitions

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	$\begin{pmatrix}1 & 0\end{pmatrix}$	Vacuous
HF/SF	2	$\mathbf{I}_{2\times 2}$	Vacuous

Avidbots Ex1016
Page 992 of 2154

Table 38.4 Vectors of contact force and moment components, also known as the wrench intensity vector, transmitted through contact i

Model	$\boldsymbol{\lambda}_i$
PwoF	(f_{in})
HF	$(f_{in}\, f_{it}\, f_{io})^{\mathrm{T}}$
SF	$(f_{in}\, f_{it}\, f_{io}\, m_{in})^{\mathrm{T}}$

where $\mathbf{M}_{\mathrm{hnd}}(\cdot)$ and $\mathbf{M}_{\mathrm{obj}}(\cdot)$ are symmetric, positive definite inertia matrices, $\boldsymbol{b}_{\mathrm{hnd}}(\cdot,\cdot)$ and $\boldsymbol{b}_{\mathrm{obj}}(\cdot,\cdot)$ are the velocity-product terms, $\boldsymbol{g}_{\mathrm{app}}$ is the force and moment applied to the object by gravity and other external sources, $\boldsymbol{\tau}_{\mathrm{app}}$ is the vector of external loads and actuator actions, and the vector $\mathbf{G}\boldsymbol{\lambda}$ is the total wrench applied to the object by the hand. The vector $\boldsymbol{\lambda}$ contains the contact force and moment components transmitted through the contacts and expressed in the contact frames. Specifically, $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^{\mathrm{T}} \cdots \boldsymbol{\lambda}_{n_c}^{\mathrm{T}}]^{\mathrm{T}}$, where

$$\boldsymbol{\lambda}_i = \mathbf{H}_i [f_{in}\, f_{it}\, f_{io}\, m_{in}\, m_{it}\, m_{io}]^{\mathrm{T}} \,.$$

The subscripts indicate one normal (n) and two tangential (t,o) components of contact force $\boldsymbol{f}$ and moment $\boldsymbol{m}$. For a SF, HF, or PwoF contact, $\boldsymbol{\lambda}_i$ is defined as in Table 38.4. Finally, it is worth noting that $\mathbf{G}_i\boldsymbol{\lambda}_i = \tilde{\mathbf{G}}_i\mathbf{H}_i^{\mathrm{T}}\boldsymbol{\lambda}_i$ is the wrench applied through contact i, where $\tilde{\mathbf{G}}_i$ and $\mathbf{H}_i$ are defined in (38.6) and (38.15). The vector $\boldsymbol{\lambda}_i$ is known as the wrench intensity vector for contact i.

Equation (38.20) represents the dynamics of the hand and object without regard for the kinematic constraints imposed by the contact models. Enforcing them, the dynamic model of the system can be written as follows

$$\begin{pmatrix} \mathbf{J}^{\mathrm{T}} \\ -\mathbf{G} \end{pmatrix} \boldsymbol{\lambda} = \begin{pmatrix} \boldsymbol{\tau} \\ \boldsymbol{g} \end{pmatrix} \tag{38.21}$$

subject to $\mathbf{J}\dot{\boldsymbol{q}} = \mathbf{G}^{\mathrm{T}}\boldsymbol{v} = \boldsymbol{v}_{\mathrm{cc}}$, where

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\tau}_{\mathrm{app}} - \mathbf{M}_{\mathrm{hnd}}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \boldsymbol{b}_{\mathrm{hnd}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \boldsymbol{g} &= \boldsymbol{g}_{\mathrm{app}} - \mathbf{M}_{\mathrm{obj}}(\boldsymbol{u})\dot{\boldsymbol{v}} - \boldsymbol{b}_{\mathrm{obj}}(\boldsymbol{u}, \boldsymbol{v}) \,. \end{aligned} \tag{38.22}$$

One should notice that the dynamic equations are closely related to the velocity kinematic model in (38.17). Specifically, just as $\mathbf{J}$ and $\mathbf{G}^{\mathrm{T}}$ transmit only selected components of contact twists, $\mathbf{J}^{\mathrm{T}}$ and $\mathbf{G}$ in (38.20) serve to transmit only the corresponding components of the contact wrenches.

When the inertia terms are negligible, as occurs during slow motion, the system is said to be quasi-static. In this case, (38.22) becomes

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\tau}_{\mathrm{app}} \,, \\ \boldsymbol{g} &= \boldsymbol{g}_{\mathrm{app}} \,, \end{aligned} \tag{38.23}$$

and does not depend on joint and object velocities. Consequently, when the grasp is in static equilibrium or moves quasi-statically, one can solve the first equation and the constraint in (38.21) independently to compute $\boldsymbol{\lambda}$, $\dot{\boldsymbol{q}}$, and $\boldsymbol{v}$. It is worth noting that such a force/velocity decoupled solution is not possible when dynamic effects are appreciable, since the first equation in (38.21) depends on the third one through (38.22).

Remark 38.2

Equation (38.21) highlights an important alternative view of the grasp matrix and the hand Jacobian. $\mathbf{G}$ can be thought of as a mapping from the transmitted contact forces and moments to the set of wrenches that the hand can apply to the object, while $\mathbf{J}^{\mathrm{T}}$ can be thought of as a mapping from the transmitted contact forces and moments to the vector of joint loads. Notice that these interpretations hold for both dynamic and quasi-static conditions.

38.2 Controllable Twists and Wrenches

In hand design and in grasp and manipulation planning, it is important to know the set of twists that can be imparted to the object by movements of the fingers, and conversely, the conditions under which the hand can prevent all possible motions of the object. The dual view is that one needs to know the set of wrenches that the hand can apply to the object and under what conditions any wrench in $\mathbb{R}^6$ can be applied through the contacts. This knowledge will be gained by studying the various subspaces associated with $\mathbf{G}$ and $\mathbf{J}$ [38.17].

The spaces, shown in Fig. 38.3, are the column spaces and null spaces of $\mathbf{G}$, $\mathbf{G}^{\mathrm{T}}$, $\mathbf{J}$, and $\mathbf{J}^{\mathrm{T}}$. Column space (also known as range) and null space will be denoted by $\mathcal{R}(\cdot)$ and $\mathcal{N}(\cdot)$, respectively. The arrows show the propagation of the various velocity and load quantities through the grasping system. For example, in the left part of Fig. 38.3 it is shown how any vector $\dot{\boldsymbol{q}} \in \mathbb{R}^{n_q}$ can be decomposed into a sum of two orthogonal vectors in $\mathcal{R}(\mathbf{J}^{\mathrm{T}})$ and in $\mathcal{N}(\mathbf{J})$ and how $\dot{\boldsymbol{q}}$ is mapped to $\mathcal{R}(\mathbf{J})$ by multiplication by $\mathbf{J}$.

It is important to recall two facts from linear algebra. First, a matrix $\mathbf{A}$ maps vectors from $\mathcal{R}(\mathbf{A}^{\mathrm{T}})$ to $\mathcal{R}(\mathbf{A})$ in a one-to-one and onto fashion, that is, the map $\mathbf{A}$ is a bijection. The generalized inverse $\mathbf{A}^{+}$ of

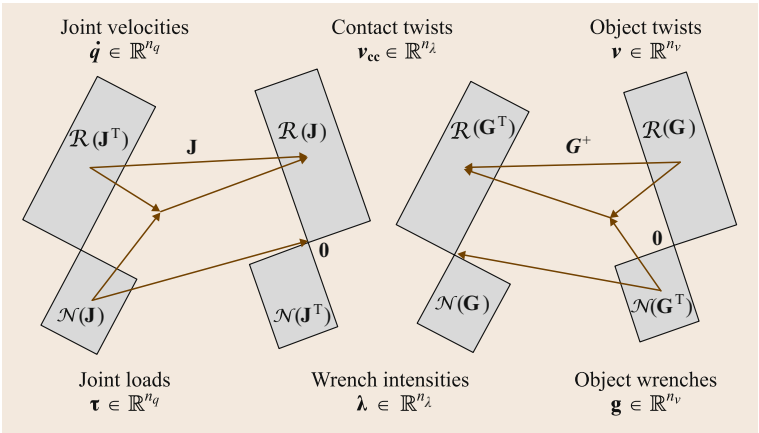


Fig. 38.3 Linear maps relating twists and wrenches of a grasping system

$\mathbf{A}$ is a bijection that maps vectors in the opposite direction [38.18]. Also, $\mathbf{A}$ maps all vectors in $\mathcal{N}(\mathbf{A})$ to zero. Finally, there is no non-trivial vector that $\mathbf{A}$ can map into $\mathcal{N}(\mathbf{A}^T)$. This implies that if $\mathcal{N}(\mathbf{G}^T)$ is non-trivial, then the hand will not be able to control all degrees of freedom of the object's motion. This is certainly true for quasi-static grasping, but when dynamics are important, they may cause the object to move along the directions in $\mathcal{N}(\mathbf{G}^T)$.

38.2.1 Grasp Classifications

The four null spaces motivate a basic classification of grasping systems defined in Table 38.5. Assuming solutions to (38.21) exist, the following force and velocity equations provide insight into the physical meaning of the various null spaces

$$\dot{q} = \mathbf{J}^+ v_{cc} + \mathbf{N}(\mathbf{J})\gamma \,, \tag{38.24}$$

$$v = (\mathbf{G}^T)^+ v_{cc} + \mathbf{N}(\mathbf{G}^T)\gamma \,, \tag{38.25}$$

$$\lambda = -\mathbf{G}^+ g + \mathbf{N}(\mathbf{G})\gamma \,, \tag{38.26}$$

$$\lambda = (\mathbf{J}^T)^+ \tau + \mathbf{N}(\mathbf{J}^T)\gamma \,. \tag{38.27}$$

In these equations, $\mathbf{A}^+$ denotes the generalized inverse, henceforth pseudoinverse, of a matrix $\mathbf{A}$, $\mathbf{N}(\mathbf{A})$ denotes a matrix whose columns form a basis for $\mathcal{N}(\mathbf{A})$, and γ is an arbitrary vector of appropriate dimension that parameterizes the solution sets. If not otherwise specified, the context will make clear if the generalized inverse is left or right.

Table 38.5 Basic grasp classes

Condition	Class	Many-to-one
$\mathcal{N}(\mathbf{J}) \neq \mathbf{0}$	Redundant	$\dot{q} \to v_{cc}$ $\tau \to \lambda$
$\mathcal{N}(\mathbf{G}^T) \neq \mathbf{0}$	Indeterminate	$v \to v_{cc}$ $g \to \lambda$
$\mathcal{N}(\mathbf{G}) \neq \mathbf{0}$	Graspable	$\lambda \to g$ $v_{cc} \to v$
$\mathcal{N}(\mathbf{J}^T) \neq \mathbf{0}$	Defective	$\lambda \to \tau$ $v_{cc} \to \dot{q}$

If the null spaces represented in the equations are non-trivial, then one immediately sees the first many-to-one mapping in Table 38.5. To see the other many-to-one mappings, and in particular the defective class, consider (38.24). It can be rewritten with v_{cc} decomposed into orthogonal components v_{rs} and v_{lns} in $\mathcal{R}(\mathbf{J})$ and $\mathcal{N}(\mathbf{J}^T)$, respectively

$$\dot{q} = \mathbf{J}^+ (v_{rs} + v_{lns}) + \mathbf{N}(\mathbf{J})\gamma \,. \tag{38.28}$$

Recall that every vector in $\mathcal{N}(\mathbf{A}^T)$ is orthogonal to every row of $\mathbf{A}^+$. Therefore $\mathbf{J}^+ v_{lns} = \mathbf{0}$. If γ and v_{rs} are fixed in (38.28), then $\dot{q}$ is unique. Thus it is clear that if $\mathcal{N}(\mathbf{J}^T)$ is non-trivial, then a subspace of twists of the hand at the contacts will map to a single joint velocity vector. Applying the same approach to the other three equations (38.25)–(38.27) yields the other many-to-one mappings listed in Table 38.5.

Equations (38.21) and (38.24)–(38.27), motivate the following definitions.

Definition 38.1 Redundant
A grasping system is said to be *redundant* if $\mathcal{N}(\mathbf{J})$ is non-trivial.

Joint velocities $\dot{q}$ in $\mathcal{N}(\mathbf{J})$ are referred to as *internal hand velocities*, since they correspond to finger motions, but do not generate motion of the hand in the constrained directions at the contact points. If the quasi-static model applies, it can be shown that these motions are not influenced by the motion of the object and vice versa.

Avidbots Ex1016
Page 994 of 2154

Definition 38.2 Indeterminate
A grasping system is said to be *indeterminate* if $\mathcal{N}(\mathbf{G}^{\mathrm{T}})$ is non-trivial.

Object twists $\boldsymbol{v}$ in $\mathcal{N}(\mathbf{G}^{\mathrm{T}})$ are called *internal object twists*, since they correspond to motions of the object, but do not cause motion of the object in the constrained directions at the contacts. If the static model applies, it can be shown that these twists cannot be controlled by finger motions.

Definition 38.3 Graspable
A grasping system is said to be *graspable* if $\mathcal{N}(\mathbf{G})$ is non-trivial.

Wrench intensities $\boldsymbol{\lambda}$ in $\mathcal{N}(\mathbf{G})$ are referred to as *internal object forces*. These wrenches are *internal* because they do not contribute to the acceleration of the object, i. e., $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$. Instead, these wrench intensities affect the tightness of the grasp. Thus, internal wrench intensities play a fundamental role in maintaining grasps that rely on friction (Sect. 38.4.2).

Definition 38.4 Defective
A grasping system is said to be *defective* if $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ is non-trivial.

Wrench intensities $\boldsymbol{\lambda}$ in $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ are called *internal hand forces*. These forces do not influence the hand joint dynamics given in (38.20). If the static model is considered, it can be easily shown that wrench intensities belonging to $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ cannot be generated by joint actions, but can be resisted by the structure of the hand.

See Example 1, Part 4, Example 2, Part 2 and Example 3, Part 3.

38.2.2 Limitations of Rigid-Body Formulation

The rigid-body dynamics equation (38.20) can be rewritten with Lagrange multipliers associated with the contact constraints as

$$\mathbf{M}_{\mathrm{dyn}} \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \dot{\boldsymbol{\nu}} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\tau} - \boldsymbol{b}_{\mathrm{hnd}} \\ \boldsymbol{\nu} - \boldsymbol{b}_{\mathrm{obj}} \\ \boldsymbol{b}_{\mathrm{c}} \end{pmatrix}, \tag{38.29}$$

where $\boldsymbol{b}_{\mathrm{c}} = [\partial(\mathbf{J}\dot{\boldsymbol{q}})/\partial \boldsymbol{q}]\dot{\boldsymbol{q}} - [\partial(\mathbf{G}\boldsymbol{\nu})/\partial \boldsymbol{u}]\dot{\boldsymbol{u}}$ and

$$\mathbf{M}_{\mathrm{dyn}} = \begin{pmatrix} \mathbf{M}_{\mathrm{hnd}} & \mathbf{0} & \mathbf{J}^{\mathrm{T}} \\ \mathbf{0} & \mathbf{M}_{\mathrm{obj}} & -\mathbf{G} \\ \mathbf{J} & -\mathbf{G}^{\mathrm{T}} & \mathbf{0} \end{pmatrix}.$$

In order for this equation to completely determine the motion of the system, it is necessary that matrix $\mathbf{M}_{\mathrm{dyn}}$ be invertible. This case is considered in detail in [38.19], where the dynamics of multi-finger manipulation is studied under the hypothesis that the hand Jacobian is full row rank, i. e., $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$. For all manipulation systems with non-invertible $\mathbf{M}_{\mathrm{dyn}}$, rigid-body dynamics fails to determine the motion and the wrench intensity vector. By observing that

$$\mathcal{N}(\mathbf{M}_{\mathrm{dyn}}) = \{(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{\nu}}, \boldsymbol{\lambda})^{\mathrm{T}} | \ddot{\boldsymbol{q}} = \mathbf{0}, \dot{\boldsymbol{\nu}} = \mathbf{0}, \boldsymbol{\lambda} \in \mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G})\},$$

the same arguments apply under quasi-static conditions defined by (38.21) and (38.23). When $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) \neq \mathbf{0}$, the rigid-body approach fails to solve the first equation in (38.21), thus leaving $\boldsymbol{\lambda}$ indeterminate.

Definition 38.5 Hyperstatic
A grasping system is said to be *hyperstatic* if

$$\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G})$$

is non-trivial.

In such systems there are internal forces (Definition 38.3) belonging to $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ that are not controllable as discussed for defective grasps. Rigid-body dynamics is not satisfactory for hyperstatic grasps, since the rigid-body assumption leads to undetermined contact wrenches [38.20].

See Example 3, Part 3.

38.2.3 Desirable Properties

For a general purpose grasping system, there are three main desirable properties: control of the object twist $\boldsymbol{\nu}$, control of object wrench $\boldsymbol{g}$, and control of the internal forces. Control of these quantities implies that the hand can deliver the desired $\boldsymbol{\nu}$ and $\boldsymbol{g}$ with specified grip pressure by the appropriate choice of joint velocities and actions. The conditions on $\mathbf{J}$ and $\mathbf{G}$ equivalent to these properties are given in Table 38.6.

We derive the conditions in two steps. First, we ignore the structure and configuration of the hand (captured in $\mathbf{J}$) by assuming that each contact point on each

Table 38.6 Desirable properties of a grasp

Task requirement	Required conditions
All wrenches possible, $\boldsymbol{g}$ All twists possible, $\boldsymbol{\nu}$	$\mathrm{rank}(\mathbf{G}) = n_{\nu}$
Control all wrenches, $\boldsymbol{g}$ Control all twists, $\boldsymbol{\nu}$	$\mathrm{rank}(\mathbf{GJ}) = \mathrm{rank}(\mathbf{G}) = n_{\nu}$
Control all internal forces	$\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$

Avidbots Ex1016
Page 995 of 2154



Fig. 38.4 The Salisbury Hand

finger can be commanded to move in any direction transmitted by the chosen contact model. An important perspective here is that $\boldsymbol{\nu}_{cc}$ is seen as the independent input variable and $\boldsymbol{\nu}$ is seen as the output. The dual interpretation is that the actuators can generate any contact force and moment in the constrained directions at each contact. Similarly, $\boldsymbol{\lambda}$ is seen as the input and $\boldsymbol{g}$ is seen as the output. The primary property of interest under this assumption is whether or not the arrangement and types of contacts on the object (captured in $\mathbf{G}$) are such that a sufficiently dexterous hand could control its fingers so as to impart any twist $\boldsymbol{\nu} \in \mathbb{R}^6$ to the object and, similarly, to apply any wrench $\boldsymbol{g} \in \mathbb{R}^6$ to the object.

All Object Twists Possible

Given a set of contact locations and types, by solving (38.19) for $\boldsymbol{\nu}$ or observing the map $\mathbf{G}$ on the right side of Fig. 38.3, one sees that the achievable object twists are those in $\mathcal{R}(\mathbf{G})$. Those in $\mathcal{N}(\mathbf{G}^{\mathrm{T}})$ cannot be achieved by any hand using the given grasp. Therefore, to achieve any object twist, we must have: $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = \mathbf{0}$, or equivalently, $\mathrm{rank}(\mathbf{G}) = n_\nu$. Any grasp with three non-collinear hard contacts or two distinct soft contacts satisfies this condition.

All Object Wrenches Possible

This case is the dual of the previous case, so we expect the same condition. From (38.21), one immediately obtains the condition $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = \mathbf{0}$, so again we have $\mathrm{rank}(\mathbf{G}) = n_\nu$.

To obtain the conditions needed to control the various quantities of interest, the structure of the hand cannot be ignored. Recall that the only achievable contact twists on the hand are in $\mathcal{R}(\mathbf{J})$, which is not necessarily equal to $\mathbb{R}^{n_\lambda}$.

Control All Object Twists

By solving (38.17) for $\boldsymbol{\nu}$, one sees that in order to cause any object twist $\boldsymbol{\nu}$ by choice of joint velocities $\dot{\boldsymbol{q}}$, we must have $\mathcal{R}(\mathbf{GJ}) = \mathbb{R}^{n_\nu}$ and $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = 0$. These conditions are equivalent to $\mathrm{rank}(\mathbf{GJ}) = \mathrm{rank}(\mathbf{G}) = n_\nu$.

Control All Object Wrenches

This property is dual to the previous one. Analysis of (38.21) yields the same conditions:

$$\mathrm{rank}(\mathbf{GJ}) = \mathrm{rank}(\mathbf{G}) = n_\nu \ .$$

Control All Internal Forces

Equation (38.20) shows that wrench intensities with no effect on object motion are those in $\mathcal{N}(\mathbf{G})$. In general, not all the internal forces may be actively controlled by joint actions. In [38.16, 21] it has been shown that all internal forces in $\mathcal{N}(\mathbf{G})$ are controllable if and only if $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$.

See Example 1, Part 5 and Example 2, Part 3.

Design Considerations of the Salisbury Hand

The Salisbury Hand in Fig. 38.4 was designed to have the smallest number of joints that would meet all the task requirements in Table 38.6. Assuming HF contacts, three non-collinear contacts is the minimum number such that $\mathrm{rank}(\mathbf{G}) = n_\nu = 6$. In this case, $\mathbf{G}$ has six rows and nine columns and the dimension of $\mathcal{N}(\mathbf{G})$ is three [38.2, 8]. The ability to control all internal forces and apply an arbitrary wrench to the object requires that $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$, so the minimum dimension of the column space of $\mathbf{J}$ is nine. To achieve this, the hand must have at least nine joints, which Salisbury implemented as three fingers, each with three revolute joints.

The intended way to execute a dexterous manipulation task with the Salisbury Hand is to grasp the object at three non-collinear points with the fingertips, forming a grasp triangle. To secure the grasp, the internal forces are controlled so that the contact points are maintained without sliding. Dexterous manipulation can be thought of as moving the fingertips to control the positions of the vertices of the grasp triangle.

Avidbots Ex1016
Page 996 of 2154

38.3 Compliant Grasps

In this section we extend the rigid-body model to include compliance. This is needed to design controllers that can implement desired compliance behaviors of a grasped object when it contacts the environment, which can increase the robustness of static grasps and dexterous manipulation tasks. It also facilitates the analysis of hands designed with flexible mechanical elements [38.22, 23] and grasp control strategies that exploit grasp *synergies* [38.24, 25]. Thanks to compliance, hands can be designed to maintain a secure grasp with fewer joints, which provides greater mechanical robustness and reduces the complexity of planning grasps.

On the other hand, reducing the number of DOFs in robotic hands demands a compliant design of the whole structure to adapt the shape of the hand to different objects and to improve robustness with respect to uncertainties [38.26]. Compliance can be passive or active. Passive compliance is due to the structural deformation of robot components, including joints, while active compliance refers to virtual elasticity of actuators, e.g., due to the proportional action of a PD joint controller, that can be actively set by changing the control parameters [38.26–29].

In the following, we extend the grasp analysis relaxing the rigid-body contact constraints to take into account both the compliance and the low number of DOFs in robotic hands.

If the hand structure is not perfectly stiff, as shown in Fig. 38.5, the actual vector of joint variables $\boldsymbol{q}$ can be different from the reference one $\boldsymbol{q}_\mathrm{r}$, given to the joint actuator controllers, and their difference is related to the joint effort $\boldsymbol{\tau}$ through a compliance matrix $\mathbf{C}_q \in \mathbb{R}^{n_q \times n_q}$ by the constitutive equation

$$\boldsymbol{q}_\mathrm{r} - \boldsymbol{q} = \mathbf{C}_q \boldsymbol{\tau} \ . \tag{38.30}$$

Note that if the hand structure is perfectly rigid, $\mathbf{C}_q = 0$ and the hand stiffness $\mathbf{K}_q = \mathbf{C}_q^{-1}$ is not defined.

It will be clear in a moment that to deal with a low number of DOFs of the hand, a compliant model of the contact must be considered. According to Definition 38.4, in case of a low number of DOFs, it is very likely that the grasp will be defective, i. e., with a non-trivial $\mathcal{N}(\mathbf{J}^\mathrm{T})$. It is worth noting that this typically happens also in power grasps [38.30] where the hand envelops the object establishing contacts even with inner limbs. In this case the hand Jacobian is a tall matrix with a non-trivial null space of its transpose.

If the system is very defective, it is very likely that the grasp will be hyperstatic according to Definition 38.5 and consequently the rigid-body model in (38.21) is under-determined and does not admit a unique solution for the contact force vector $\boldsymbol{\lambda}$ as discussed in Sect. 38.2.2. Note that computing the force distribution $\boldsymbol{\lambda}$ is crucial in grasp analysis since it allows one to evaluate if the contact constraints are fulfilled, and consequently, if the grasp will be maintained.

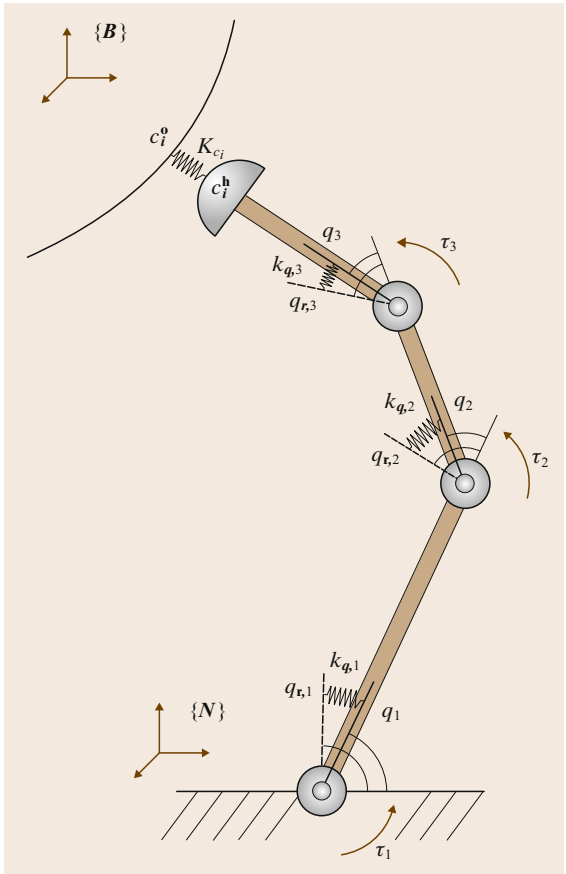


Fig. 38.5 Compliant joints and compliant contacts, main definitions

The force distribution problem in hyperstatic grasps is an under-determined problem of statics under the assumption of rigid contacts of (38.17). To solve the problem, we need to enrich the model with more information on the contact forces. A possibile solution is to substitute the rigid-body kinematic constraint (38.19) with a compliant model of the contact interaction [38.21], obtained introducing a set of springs between the contact points on the hand $\boldsymbol{c}^\mathrm{h}$ and on the object $\boldsymbol{c}^\mathrm{o}$ as shown in Fig. 38.5

$$\mathbf{C}_\mathrm{c} \boldsymbol{\lambda} = \boldsymbol{c}^\mathrm{h} - \boldsymbol{c}^\mathrm{o} \ , \tag{38.31}$$

Avidbots Ex1016
Page 997 of 2154

where $\mathbf{C}_c \in \mathbb{R}^{n_\lambda \times n_\lambda}$ is the contact compliance matrix which is symmetric and positive definite. The contact stiffness matrix is defined as the reciprocal of the contact compliance matrix $\mathbf{K}_c = \mathbf{C}_c^{-1}$.

The analysis is presented in a quasi-static framework: from an equilibrium reference configuration $\boldsymbol{q}_{r,0}$, $\boldsymbol{q}_0$, $\boldsymbol{\tau}_0$, $\boldsymbol{\lambda}_0$, $\boldsymbol{u}_0$ and $\boldsymbol{g}_0$, a small input perturbation is applied to the joint references $\boldsymbol{q}_{r,0} + \Delta\boldsymbol{q}_r$ and to the external load $\boldsymbol{g}_0 + \Delta\boldsymbol{g}$ which moves the grasp system to a new equilibrium configuration whose linear approximation is represented by $\boldsymbol{q}_0 + \Delta\boldsymbol{q}$, $\boldsymbol{\tau}_0 + \Delta\boldsymbol{\tau}$, $\boldsymbol{\lambda}_0 + \Delta\boldsymbol{\lambda}$, $\boldsymbol{u}_0 + \Delta\boldsymbol{u}$.

In the following, for the sake of simplicity, it is convenient to refer $\mathbf{G}$ and $\mathbf{J}$ matrices to the object reference frame $\{\boldsymbol{B}\}$ and not to the inertial frame $\{\boldsymbol{N}\}$ as previously described. Note that this only applies to this section on compliance analysis, while the rest of this chapter still refers to matrices $\mathbf{G}$ and $\mathbf{J}$ as described in Sect. 38.1.1.

Let $\mathbf{R}_b \in \mathbb{R}^{3\times3}$ represent the orientation of the object frame $\{\boldsymbol{B}\}$ with respect to the inertial frame. Then the object twist referred to $\{\boldsymbol{B}\}$ is given as

$$\boldsymbol{v}_{i,\mathrm{obj}} = \overline{\mathbf{R}}_b^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}_{i,\mathrm{obj}}^{\mathrm{N}} \\ \boldsymbol{\omega}_{\mathrm{obj}}^{\mathrm{N}} \end{pmatrix}, \tag{38.32}$$

where

$$\overline{\mathbf{R}}_b = \mathrm{Blockdiag}(\mathbf{R}_b, \mathbf{R}_b) = \begin{pmatrix} \mathbf{R}_b & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_b \end{pmatrix} \in \mathbb{R}^{6\times6}.$$

Then, substituting $\mathbf{P}_i^{\mathrm{T}}\boldsymbol{v}$ from (38.2) into (38.32) yields the partial grasp matrix $\tilde{\mathbf{G}}_i^{\mathrm{T}} \in \mathbb{R}^{6\times6}$, that maps the object twist from $\{\boldsymbol{N}\}$ to contact frame $\{\boldsymbol{C}\}_i$ referred to $\{\boldsymbol{B}\}$.

Similarly, the hand twists can be expressed to the object frame $\{\boldsymbol{B}\}$

$$\boldsymbol{v}_{i,\mathrm{hnd}} = \overline{\mathbf{R}}_b^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}_{i,\mathrm{hnd}}^{\mathrm{N}} \\ \boldsymbol{\omega}_{i,\mathrm{hnd}}^{\mathrm{N}} \end{pmatrix} \tag{38.33}$$

and combining (38.33) and (38.7) yields the partial hand Jacobian $\tilde{\mathbf{J}}_i \in \mathbb{R}^{6\times n_q}$, which relates the joint velocities to the contact twists on the hand expressed with respect to $\{\boldsymbol{B}\}$.

Table 38.7 gives the definitions of the selection matrices for the three contact models when object and hand twists are referred to frame $\{\boldsymbol{B}\}$. In the planar case, as far as contact constraint is concerned, equation (38.15) holds with $\mathbf{H}_{i\mathrm{F}}$ and $\mathbf{H}_{i\mathrm{M}}$ defined in Table 38.8.

The contact force variation $\Delta\boldsymbol{\lambda}$ from the equilibrium configuration is then related to the relative displacement between the object and the fingers at the contact points as

$$\mathbf{C}_c\Delta\boldsymbol{\lambda} = (\mathbf{J}\Delta\boldsymbol{q} - \mathbf{G}^{\mathrm{T}}\Delta\boldsymbol{u}). \tag{38.34}$$

Table 38.7 Selection matrices for three contact models, when the object twists are expressed with respect to $\{\boldsymbol{B}\}$

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	$\hat{\boldsymbol{n}}_i^{b\mathrm{T}}$	Vacuous
HF	3	$\left[\hat{\boldsymbol{n}}_i^b, \hat{\boldsymbol{t}}_i^b, \hat{\boldsymbol{o}}_i^b\right]^{\mathrm{T}}$	Vacuous
SF	4	$\left[\hat{\boldsymbol{n}}_i^b, \hat{\boldsymbol{t}}_i^b, \hat{\boldsymbol{o}}_i^b\right]^{\mathrm{T}}$	$\hat{\boldsymbol{n}}_i^{b\mathrm{T}}$

$\hat{\boldsymbol{n}}_i^b$ is the normal unit vector at the contact i expressed with respect to $\{\boldsymbol{B}\}$ reference system.

Table 38.8 Selection matrices for the planar simplified case, when the object twists are expressed with respect to $\{\boldsymbol{B}\}$

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	$\hat{\boldsymbol{n}}_i^{b\mathrm{T}}$	Vacuous
HF/SF	2	$\left[\hat{\boldsymbol{n}}_i^b, \hat{\boldsymbol{t}}_i^b\right]^{\mathrm{T}}$	Vacuous

By assuming that the perturbed configuration is sufficiently near to the reference one, the following linearized relationships can be found for $\boldsymbol{g}$, and $\boldsymbol{\tau}$

$$\Delta\boldsymbol{g} = -\mathbf{G}\Delta\boldsymbol{\lambda} \tag{38.35}$$

$$\Delta\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\Delta\boldsymbol{\lambda} + \mathbf{K}_{J,q}\Delta\boldsymbol{q} + \mathbf{K}_{J,u}\Delta\boldsymbol{u} \tag{38.36}$$

where

$$\mathbf{K}_{J,q} = \frac{\partial\mathbf{J}\boldsymbol{\lambda}_0}{\partial\boldsymbol{q}} \quad \text{and} \quad \mathbf{K}_{J,u} = \frac{\partial\mathbf{J}\boldsymbol{\lambda}_0}{\partial\boldsymbol{u}}$$

are the variation of the hand Jacobian with respect to $\boldsymbol{q}$ and $\boldsymbol{u}$ variations, respectively.

Remark 38.3

In (38.35) and (38.36) both the grasp matrix and hand Jacobian are expressed with respect to the object reference frame, and by neglecting rolling between the fingers and the object at the contact points, $\mathbf{G}$ becomes constant, while $\mathbf{J}(\boldsymbol{q},\boldsymbol{u})$, in general, depends on both hand and object configurations. Matrix $\mathbf{K}_{J,q}$ represent the variability of $\mathbf{J}(\boldsymbol{q},\boldsymbol{u})$ matrix with respect to hand and object configurations.

Matrices $\mathbf{K}_{J,q} \in \mathbb{R}^{n_q\times n_q}$ and $\mathbf{K}_{J,u} \in \mathbb{R}^{n_q\times n_v}$ are usually referred to as *geometric stiffness matrices* [38.28]. Furthermore, it can be verified that the matrix $\mathbf{K}_{J,q}$ is symmetric [38.27].

By substituting (38.36) in (38.30) we get

$$\mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{J}^{\mathrm{T}}\Delta\boldsymbol{\lambda} = \mathbf{J}_{\mathrm{R}}\Delta\boldsymbol{q}_r - \mathbf{J}\Delta\boldsymbol{q} - \mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{K}_{J,u}\Delta\boldsymbol{u}, \tag{38.37}$$

where

$$\mathbf{J}_{\mathrm{R}} = \mathbf{J}\left(\mathbf{I} + \mathbf{C}_q\mathbf{K}_{J,q}\right)^{-1}.$$

Summing (38.34) and (38.37) the following expression can be found for the contact forces displacement $\Delta\boldsymbol{\lambda}$

$$\Delta\boldsymbol{\lambda} = \mathbf{K}_{c,e}\left(\mathbf{J}_{\mathrm{R}}\Delta\boldsymbol{q}_{\mathrm{r}} - \mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\Delta\boldsymbol{u}\right) \tag{38.38}$$

where

$$\mathbf{G}_{\mathrm{R}}^{\mathrm{T}} = \mathbf{G}^{\mathrm{T}} + \mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{K}_{J,u}$$

and

$$\mathbf{K}_{c,e} = \left(\mathbf{C}_{\mathrm{c}} + \mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{J}^{\mathrm{T}}\right)^{-1} . \tag{38.39}$$

Matrix $\mathbf{K}_{c,e}$ represents the equivalent contact stiffness, that takes into account both the joint and the contact compliance. If the geometrical terms are neglected, i. e., $\mathbf{K}_{J,q} = 0$ and $\mathbf{K}_{J,u} = 0$, the classical expression $\mathbf{K}_{c,e} = \left(\mathbf{C}_{\mathrm{c}} + \mathbf{J}\mathbf{C}_q\mathbf{J}^{\mathrm{T}}\right)$ for the equivalent contact stiffness matrix can be found [38.28].

By substituting (38.38) in (38.35), the object displacement can be evaluated as a function of the small input perturbations $\Delta\boldsymbol{q}_{\mathrm{r}}$ and $\Delta\boldsymbol{g}$

$$\Delta\boldsymbol{u} = \left(\mathbf{G}\mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\right)^{-1}\left(\mathbf{G}\mathbf{K}_{c,e}\mathbf{J}_{\mathrm{R}}\Delta\boldsymbol{q}_{\mathrm{r}} + \Delta\boldsymbol{g}\right) . \tag{38.40}$$

When $\Delta\boldsymbol{q}_{\mathrm{r}} = 0$, (38.40) can be rewritten as

$$\Delta\boldsymbol{g} = \mathbf{K}\Delta\boldsymbol{u} ,$$
$$\text{with } \mathbf{K} = \mathbf{G}\mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}} ,$$

where the term that multiplies the external wrench variation $\Delta\boldsymbol{g}$ represents the reciprocal of the *grasp stiffness matrix* $\mathbf{K}$. The grasp stiffness evaluates ability of the robotic grasp to resist to external load variations applied to the object.

Regarding the force distribution $\Delta\boldsymbol{\lambda}$, by substituting (38.40) into (38.38), the variation of the contact forces is evaluated as

$$\Delta\boldsymbol{\lambda} = \mathbf{G}_g^{+}\Delta\boldsymbol{g} + \mathbf{P}\Delta\boldsymbol{q}_{\mathrm{r}} , \tag{38.41}$$

where

$$\mathbf{G}_g^{+} = \mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\left(\mathbf{G}\mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\right)^{-1} ,$$
$$\mathbf{P} = \left(\mathbf{I} - \mathbf{G}_g^{+}\mathbf{G}\right)\mathbf{K}_{c,e}\mathbf{J}_{\mathrm{R}} .$$

Matrix $\mathbf{G}_g^{+}$ is a right pseudo-inverse of grasp matrix $\mathbf{G}$ that takes into account both the geometrical effects and the hand and contact stiffness. Matrix $\mathbf{P}$ maps the reference joint variables $\Delta\boldsymbol{q}_{\mathrm{r}}$ to the contact force variation $\Delta\boldsymbol{\lambda}$.

It is worth noting that $\left(\mathbf{I} - \mathbf{G}_g^{+}\mathbf{G}\right)$ is a projector onto the null space of $\mathbf{G}$ and then each contact force variation $\Delta\boldsymbol{\lambda}_h = \mathbf{P}\Delta\boldsymbol{q}_{\mathrm{r}}$, produced by modifying joint reference values, satisfies the equation

$$\mathbf{G}\Delta\boldsymbol{\lambda}_h = 0 ,$$

and then belongs to the internal force subspace.

Summarizing, compliant grasps have been analyzed with a linearized quasi-static model and the main relationships mapping joint references, the controlled inputs, and external disturbances, the wrenches, onto object motions and contact forces have been evaluated in (38.40) and (38.41), respectively.

38.4 Restraint Analysis

The most fundamental requirements in grasping and dexterous manipulation are the abilities to hold an object and control its position and orientation relative to the palm of the hand. The two most useful characterizations of grasp restraint are *force closure* and *form closure*. These names were in use as early as 1876 in the field of machine design to distinguish between joints that required an external force to maintain contact, and those that did not [38.31]. For example, some water wheels had a cylindrical axle that was laid in a horizontal semi-cylindrical groove split on either side of the wheel. During operation, the weight of the wheel acted to *close* the groove-axle contacts, hence the term *force closure*. By contrast, if the grooves were replaced by cylindrical holes just long enough to accept the axle, then the contacts would be closed by the geometry (even if the direction of the gravitational force was reversed), hence the term *form closure*.

When applied to grasping, form and force closure have the following interpretations. Assume that a hand grasping an object has its joint angles locked and its palm fixed in space. Then the grasp has *form closure*, or the object is *form-closed*, if it is impossible to move the object, even infinitesimally. Under the same conditions, the grasp has *force closure*, or the object is *force-closed*, if for any non-contact wrench experienced by the object, contact wrench intensities exist that satisfy (38.20) and are consistent with the constraints imposed by the friction models applicable at the contact points. Notice that all form closure grasps are also force closure grasps. When under form closure, the object cannot move even infinitesimally relative to the hand, regardless of the non-contact wrench. Therefore, the hand maintains the object in equilibrium for any external wrench, which is the force closure requirement.

Avidbots Ex1016
Page 999 of 2154

Roughly speaking, form closure occurs when the palm and fingers wrap around the object forming a cage with no *wiggle room* such as the grasp shown in Fig. 38.6. This kind of grasp is also called a *power grasp* [38.32] or *enveloping grasp* [38.33]. However, force closure is possible with fewer contacts, as shown in Fig. 38.7, but in this case, force closure requires the ability to control internal forces. It is also possible for a grasp to have partial form closure, indicating that only a subset of the possible degrees of freedom are restrained [38.34]. An example of such a grasp is shown in Fig. 38.8. In this grasp, fingertip placement between the ridges around the periphery of the bottle cap provides form closure against relative rotation about the axis of the helix of the threads and also against translation perpendicular to that axis, but the other three degrees of freedom are restrained through force closure. Strictly speaking, given a grasp of a real object by a human hand it is impossible to prevent relative motion of the object with respect to the palm due to the compliance of the hand and object. Preventing all motion is possible only if the contacting bodies are rigid, as is assumed in most mathematical models employed in grasp analysis.



Fig. 38.6 The palm and fingers combine to create a very secure form closure grasp of a router



Fig. 38.7 Hand and wrist of the tendon driven DLR Hand Arm System grasping a tool [38.7]. This grasp has a force closure grasp appropriate for dexterous manipulation. (DLR Hand Arm System, Photo courtesy of DLR 2011)

38.4.1 Form Closure

To make the notion of form closure precise, introduce a signed gap function denoted by $\psi_i(\boldsymbol{u}, \boldsymbol{q})$ at each of the n_c contact points between the object and the hand. The gap function is zero at each contact, becomes positive if contact breaks, and negative if penetration occurs. The gap function can be thought of as distance between the contact points. In general, this function is dependent on the shapes of the contacting bodies. Let $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{q}}$ represent the configurations of the object and hand for a given grasp; then

$$\psi_i(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}}) = \boldsymbol{0} \quad \forall\, i = 1, \ldots, n_c\,. \tag{38.42}$$

The form closure condition can now be stated in terms of a differential change $\mathrm{d}\boldsymbol{u}$ of $\bar{\boldsymbol{u}}$:

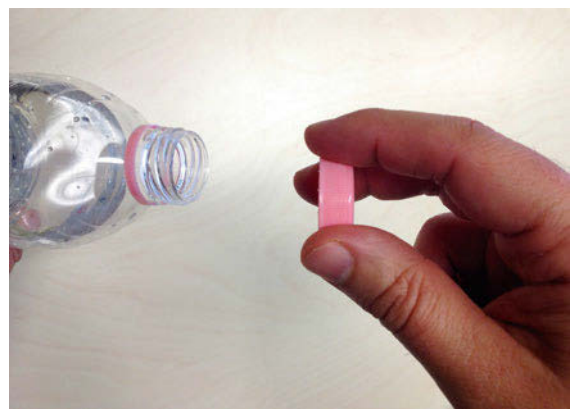


Fig. 38.8 In the grasp depicted, contact with the ridges on the bottle cap create partial form closure in the direction of cap rotation (when screwing it in) and also in the directions of translation perpendicular to the axis of rotation. To achieve complete control over the cap, the grasp achieves force closure over the other three degrees of freedom

Definition 38.6
A grasp $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})$ has *form closure* if and only if the following implication holds

$$\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}}) \geq \mathbf{0} \Rightarrow \mathrm{d}\boldsymbol{u} = \mathbf{0}\,, \tag{38.43}$$

where $\boldsymbol{\psi}$ is the n_c-dimensional vector of gap functions with i-th component equal to $\psi_i(\boldsymbol{u}, \boldsymbol{q})$. By definition, inequalities between vectors imply that the inequality is applied between corresponding components of the vectors.

Expanding the gap function vector in a Taylor series about $\bar{\boldsymbol{u}}$, yields infinitesimal form closure tests of various orders. Let ${}^{\beta}\boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q}), \beta = 1, 2, 3, \ldots,$ denote the Taylor series approximation truncated after the terms of order β in $\mathrm{d}\boldsymbol{u}$. From (38.42), it follows that the first-order approximation is

$${}^{1}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}}) = \frac{\partial \boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q})}{\partial \boldsymbol{u}}|_{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})} \mathrm{d}\boldsymbol{u}\,,$$

where $\frac{\partial \boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q})}{\partial \boldsymbol{u}}|_{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})}$ denotes the partial derivative of $\boldsymbol{\psi}$ with respect to $\boldsymbol{u}$ evaluated at $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})$. Replacing $\boldsymbol{\psi}$ with its approximation of order β in (38.43) implies three relevant cases of order β:

1. If there exists $\mathrm{d}\boldsymbol{u}$ such that ${}^{\beta}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}})$ has at least one strictly positive component, then the grasp does not have form closure of order β.
2. If for every non-zero $\mathrm{d}\boldsymbol{u}$, ${}^{\beta}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}})$ has at least one strictly negative component, then the grasp has form closure of order β.
3. If neither case i) nor case ii) applies for all ${}^{\eta}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}})\ \forall\ \eta \leq \beta$, then higher-order analysis is required to determine the existence of form closure.

Figure 38.9 illustrates form closure concepts using several planar grasps of gray objects by fingers shown as black disks. The concepts are identical for grasps of three-dimensional objects, but are more clearly illustrated in the plane. The grasp on the left has first-order form closure. Note that first-order form closure only involves the first derivatives of the distance functions. This implies that the only relevant geometry in first-order form closure are the locations of the contacts and the directions of the contact normals. The grasp in the center has form closure of higher order, with the specific order depending on the degrees of the curves defining the surfaces of the object and fingers in the neighborhoods of the contacts [38.35]. Second-order form closure analysis depends on the curvatures of the two contacting bodies in addition to the geometric information used to analyze first-order form closure. The grasp on the right does not have form closure of any order, because the object can translate horizontally and rotate about its center.

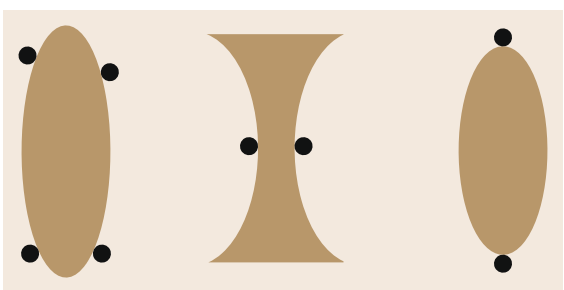


Fig. 38.9 Three planar grasps; two with form closure of different orders and one without form closure

First-Order Form Closure

First-order form closure exists if and only if the following implication holds

$$\frac{\partial \boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q})}{\partial \boldsymbol{u}}|_{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})} \mathrm{d}\boldsymbol{u} \geq \mathbf{0} \Rightarrow \mathrm{d}\boldsymbol{u} = \mathbf{0}\,.$$

The first-order form closure condition can be written in terms of the object twist $\boldsymbol{\nu}$

$$\mathbf{G}_{\mathrm{n}}^{\mathrm{T}} \boldsymbol{\nu} \geq \mathbf{0} \Rightarrow \boldsymbol{\nu} = \mathbf{0}\,, \tag{38.44}$$

where $\mathbf{G}_{\mathrm{n}}^{\mathrm{T}} = \partial \boldsymbol{\psi} / \partial \boldsymbol{u} \mathbf{V} \in \mathbb{R}^{n_c \times 6}$. Recall that $\mathbf{V}$ is the kinematic map defined in (38.1). Also notice that $\mathbf{G}_{\mathrm{n}}$ is the grasp matrix, when all contact points are frictionless.

Because the gap functions only quantify distances, the product $\mathbf{G}_{\mathrm{n}}^{\mathrm{T}} \boldsymbol{\nu}$ is the vector of normal components of the instantaneous velocities of the object at the contact points (which must be non-negative to prevent interpenetration). This in turn implies that the grasp matrix is the one that would result from the assumption that all contacts are of the type PwoF.

An equivalent condition in terms of the contact wrench intensity vector $\boldsymbol{\lambda}_{\mathrm{n}} \in \mathbb{R}^{n_c}$ can be stated as follows. A grasp has first-order form closure if and only if

$$\left.\begin{array}{l} \mathbf{G}_{\mathrm{n}} \boldsymbol{\lambda}_{\mathrm{n}} = -\boldsymbol{g} \\ \boldsymbol{\lambda}_{\mathrm{n}} \geq \mathbf{0} \end{array}\right\} \forall\ \boldsymbol{g} \in \mathbb{R}^6\,. \tag{38.45}$$

The physical interpretation of this condition is that equilibrium can be maintained under the assumption that the contacts are frictionless. Note that the components of $\boldsymbol{\lambda}_{\mathrm{n}}$ are the magnitudes of the normal components of the contact forces. The subscript $(.)_{\mathrm{n}}$ is used to emphasize that $\boldsymbol{\lambda}_{\mathrm{n}}$ contains no other force or moment components.

Avidbots Ex1016
Page 1001 of 2154

Since $\boldsymbol{g}$ must be in the range of $\mathbf{G}_n$ for equilibrium to be satisfied, and since $\boldsymbol{g}$ is an arbitrary element of $\mathbb{R}^6$, then in order for condition (38.45) to be satisfied, the rank of $\mathbf{G}_n$ must be six. Assuming $\text{rank}(\mathbf{G}_n) = 6$, another equivalent mathematical statement of first-order form closure is: there exists $\boldsymbol{\lambda}_n$ such that the following two conditions hold [38.36]

$$\mathbf{G}_n\boldsymbol{\lambda}_n = \mathbf{0}\,,$$
$$\boldsymbol{\lambda}_n > \mathbf{0}\,. \tag{38.46}$$

This means that there exists a set of strictly compressive normal contact forces in the null space of $\mathbf{G}_n$. In other words, one can squeeze the object as tightly as desired while maintaining equilibrium. A second interpretation of this condition is that the non-negative span of the columns of $\mathbf{G}_n$ must equal $\mathbb{R}^6$. Equivalently, the convex hull of the columns of $\mathbf{G}_n$ must strictly contain the origin of $\mathbb{R}^6$. As will be seen, the span and hull interpretations will provide a conceptual link called *frictional form closure* that lies between form closure and force closure.

The duality of conditions (38.44) and (38.45) can be seen clearly by examining the set of wrenches that can be applied by frictionless contacts and the corresponding set of possible object twists. For this discussion, it is useful to give definitions of cones and their duals.

Definition 38.7
A *cone* C is a set of vectors $\boldsymbol{\varsigma}$ such that for every $\boldsymbol{\varsigma}$ in C, every non-negative scalar multiple of $\boldsymbol{\varsigma}$ is also in C.

Equivalently, a cone is a set of vectors closed under addition and non-negative scalar multiplication.

Definition 38.8
Given a cone C with elements $\boldsymbol{\varsigma}$, the *dual cone* C^* with elements $\boldsymbol{\varsigma}^*$, is the set of vectors such that the dot product of $\boldsymbol{\varsigma}^*$ with each vector in C is non-negative. Mathematically

$$C^* = \{\boldsymbol{\varsigma}^* | \boldsymbol{\varsigma}^{\mathrm{T}}\boldsymbol{\varsigma}^* \geq 0, \forall \boldsymbol{\varsigma} \in C\}\,. \tag{38.47}$$

See Example 4.

First-Order Form Closure Requirements

Several useful necessary conditions for form closure are known. In 1897 *Somov* proved that at least seven contacts are necessary to form close a rigid object with six degrees of freedom [38.37, 38]. *Lakshminarayana* generalized this to prove that $n_\nu + 1$ contacts are necessary to form close an object with n_ν degrees of freedom [38.34] (based on *Goldman* and *Tucker* 1956 [38.39]) (Table 38.9). This led to the definition of partial form closure that was mentioned above in the discussion of the hand grasping the bottle cap. *Markenscoff* and *Papadimitriou* determined a tight upper bound, showing that for all objects whose surfaces are not surfaces of revolution, at most $n_\nu + 1$ contacts are necessary [38.40]. Form closure is impossible to achieve for surfaces of revolution.

To emphasize the fact that $n_\nu + 1$ contacts are necessary and *not* sufficient, consider grasping a cube with seven or more points of contact. If all contacts are on one face, then clearly, the grasp does not have form closure.

First-Order Form Closure Tests

Because form closure grasps are very secure, it is desirable to design or synthesize such grasps. To do this, one needs a way to test candidate grasps for form closure, and rank them, so that the *best* grasp can be chosen. One reasonable measure of form closure can be derived from the geometric interpretation of the condition (38.46). The null space constraint and the positivity of $\boldsymbol{\lambda}_n$ represent the addition of the columns of $\mathbf{G}_n$ scaled by the components of $\boldsymbol{\lambda}_n$. Any choice of $\boldsymbol{\lambda}_n$ closing this loop is in $\mathcal{N}(\mathbf{G}_n)$. For a given loop, if the magnitude of the smallest component of $\boldsymbol{\lambda}_n$ is positive, then the grasp has form closure, otherwise it does not. Let us denote this smallest component by d. Since such a loop, and hence d, can be scaled arbitrarily, $\boldsymbol{\lambda}_n$ must be bounded.

After verifying that $\mathbf{G}_n$ has full row rank, a quantitative form closure test based on the above observations can be formulated as a linear program (LP) in the unknowns d and $\boldsymbol{\lambda}_n$ as follows

LP1: maximize: d (38.48)
subject to: $\mathbf{G}_n\boldsymbol{\lambda}_n = \mathbf{0}$ (38.49)
$\mathbf{I}\boldsymbol{\lambda}_n - \mathbf{1}d \geq \mathbf{0}$ (38.50)
$d \geq 0$ (38.51)
$\mathbf{1}^{\mathrm{T}}\boldsymbol{\lambda}_n \leq n_c\,,$ (38.52)

where $\mathbf{I} \in \mathbb{R}^{n_c \times n_c}$ is the identity matrix and $\mathbf{1} \in \mathbb{R}^{n_c}$ is a vector with all components equal to 1. The last inequality is designed to prevent this LP from becoming unbounded. A typical LP solution algorithm determines

Table 38.9 Minimum number of contacts n_c required to form close an object with n_ν degrees of freedom

n_ν	n_c
3 (planar grasp)	4
6 (spatial grasp)	7
n_ν (general)	$n_\nu + 1$

infeasibility or unboundedness of the constraints in Phase I of the algorithm, and considers the result before attempting to calculate an optimal value [38.41]. If LP1 is infeasible, or if the optimal value d^* is zero, then the grasp is not form closed.

The quantitative form closure test (38.48)–(38.52) has $n_c + 8$ constraints and $n_c + 1$ unknowns. For a typical grasp with $n_c < 10$, this is a small linear program that can be solved very quickly using the simplex method. However, one should note that d^* is dependent on the choice of units used when forming $\mathbf{G}_n$. It would be advisable to non-dimensionalize the components of the wrenches to avoid dependence of the optimal d on one's choice of units. This could be done by dividing the first three rows of $\mathbf{G}_n$ by a characteristic force and the last three rows by a characteristic moment. If one desires a binary test, LP1 can be converted into one by dropping the last constraint (38.52) and applying only Phase I of the simplex algorithm.

In summary, quantitative form closure testing is a two-step process:

Form Closure Test.

1. Compute rank($\mathbf{G}_n$).
   a) If rank($\mathbf{G}_n$) $\neq n_v$, then form closure does not exist. Stop.
   b) If rank($\mathbf{G}_n$) $= n_v$, continue.
2. Solve LP1.
   a) If $d^* = 0$, then form closure does not exist.
   b) If $d^* > 0$, then form closure exists and $d^* n_c$ is a crude measure of how far the grasp is from losing form closure.

See Example 5, Part 1.

Variations of the Test. If the rank test fails, then the grasp could have partial form closure over as many as rank($\mathbf{G}_n$) degrees of freedom. If one desires to test this, then LP1 must be solved using a new $\mathbf{G}_n$ formed by retaining only the rows corresponding to the degrees of freedom for which partial form closure is to be tested. If $d^* > 0$, then partial form closure exists.

A second variation arises when one knows in advance that the object is already partially constrained. For example, in the case of a steering wheel, the driver knows, that relative to her, the steering wheel has only one degree of freedom. A form closure grasp suitable for driving would be required only to restraint the wheel's rotation about the steering column. In general, assume that the object is constrained by a set of bilateral constraints, which can be written as

$$\mathbf{B}^{\mathrm{T}} \boldsymbol{v} = \mathbf{0} \,. \tag{38.53}$$

With these additional constraints, the form closure property can be expressed as follows

$$\left.\begin{aligned} \mathbf{G}_n^{\mathrm{T}} \boldsymbol{v} &\geq \mathbf{0} \\ \mathbf{B}^{\mathrm{T}} \boldsymbol{v} &= \mathbf{0} \end{aligned}\right\} \Rightarrow \boldsymbol{v} = \mathbf{0} \,, \tag{38.54}$$

which can be shown to be equivalent to

$$\overline{\mathbf{G}}_n^{\mathrm{T}} \boldsymbol{\rho} \geq \mathbf{0} \Rightarrow \boldsymbol{\rho} = \mathbf{0} \,, \tag{38.55}$$

where $\boldsymbol{\rho}$ is an arbitrary vector with length equal to the dimension of the null space of $\mathbf{B}^{\mathrm{T}}$ and $\overline{\mathbf{G}}_n = \mathbf{A}_B \mathbf{G}_n$, where $\mathbf{A}_B$ is an annihilator of the column space of $\mathbf{B}$. One possible way to construct this annihilator is $\mathbf{A}_B = \left(\mathbf{N}\left(\mathbf{B}^{\mathrm{T}}\right)\right)^{\mathrm{T}}$, where $\mathbf{N}\left(\mathbf{B}^{\mathrm{T}}\right)$ is a matrix whose columns form a basis for the null space of $\mathbf{B}^{\mathrm{T}}$. Also note that $N(\mathbf{B}^{\mathrm{T}})\boldsymbol{\rho}$ is a possible twist of the object consistent with the bilateral constraints (i. e., the twists that must be eliminated by the unilateral constraints).

The form closure condition for an object partially constrained by bilateral constraints can also be stated in terms of wrenches

$$\begin{aligned} \overline{\mathbf{G}}_n \boldsymbol{\lambda}_n &= \mathbf{0} \,, \\ \boldsymbol{\lambda}_n &> \mathbf{0} \,. \end{aligned} \tag{38.56}$$

Notice that twist conditions (38.44) and (38.55) and wrench conditions (38.46) and (38.56) are formally analogous, and therefore, the quantitative form closure test can be applied by substituting $\overline{\mathbf{G}}_n$ for $\mathbf{G}_n$ and $(n_v - \mathrm{rank}(\mathbf{B}))$ for n_v. For detailed information on the derivations of the constrained form closure conditions, please refer to [38.42].

See Example 5, Part 2.

Monotonicity. In grasp synthesis, sometimes it is desirable for the grasp metric to increase monotonically with the number of contact points, the intuition being, that adding contacts to a stable grasp will make it more stable. The solution d^* does not have this property. However `formClosure.m`, a Matlab function which is available at [38.43], returns $d^* n_c$ as the metric, because it nearly maintains monotonicity when adding random contacts to random grasps. This can be demonstrated by running `test_monotonicity.m`.

One can design metrics that are monotonic with norms on the set of wrenches that the hand can apply. Let the polytope $\mathcal{G} = \{\boldsymbol{g} | \boldsymbol{g} = \mathbf{G}_n \boldsymbol{\lambda}_n, \mathbf{0} \leq \boldsymbol{\lambda}_n \leq \mathbf{1}\}$ denote the wrench space, which is the set of wrenches that can be applied by the hand if the contact forces magnitudes are limited to one. A metric proposed by *Ferrari* and *Canny* [38.44] is the radius of the largest sphere in $\mathcal{G}$ with origin at the origin of the wrench space. The radius is greater than 0 if and only if the grasp has form

Avidbots Ex1016
Page 1003 of 2154

closure. In addition, adding a contact point will always produce a new $\mathcal{G}$ that is a superset of the original $\mathcal{G}$. Thus the radius of the sphere cannot reduce as a result. Hence the monotonicity property holds.

See Example 5, Part 3.

Planar Simplifications

In the planar case, *Nguyen* [38.45] developed a graphical qualitative test for form closure. Figure 38.10 shows a form closure grasp with four contacts. To test form closure one partitions the normals into two groups of two. Let C_1 be the non-negative span of two normals in one pair and C_2 be the non-negative span of the other pair. A grasp has form closure if and only if C_1 and C_2 or $-C_1$ and $-C_2$ *see each other* for any pairings. Two cones *see each other* if the open line segment defined by the vertices of the cones lies in the interior of both cones. In the presence of more than four contacts, if any set of four contacts satisfies this condition, then the grasp has form closure.

Notice that this graphical test can be difficult to execute for grasps with more than four contacts. Also, it does not extend to grasps of three-dimensional (3-D) objects and does not provide a closure measure.

38.4.2 Force Closure

A grasp has force closure, or is force-closed, if the grasp can be maintained in the face of any object wrench. Force closure is similar to form closure, but relaxed to allow friction forces to help balance the object wrench. A benefit of including friction in the analysis is the reduction in the number of contact points needed for closure. A three-dimensional object with six degrees of freedom requires seven contacts for form closure, but for force closure, only two contacts are needed if they are modeled as soft fingers, and only three (non-collinear) contacts are needed if they are modeled as hard fingers.

Force closure relies on the ability of the hand to squeeze arbitrarily tightly in order to compensate for large applied wrenches that can only be resisted by friction. Figure 38.15 shows a grasped polygon. Consider applying a wrench to the object that is a pure force acting upward along the $\boldsymbol{y}$-axis of the inertial frame. It seems intuitive that if there is enough friction, the hand will be able to squeeze the object with friction forces preventing the object's upward escape. Also, as the applied force increases in magnitude, the magnitude of the squeezing force will have to increase accordingly.

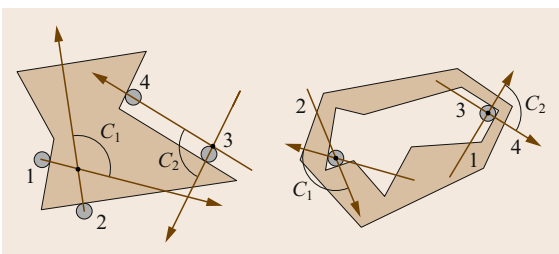


Fig. 38.10 Planar grasps with first-order form closure

Since force closure is dependent on the friction models, common models will be introduced before giving formal definitions of force closure.

Friction Models

Recall the components of force and moment transmitted through contact i under the various contact models given earlier in Table 38.4. At contact point i, the friction law imposes constraints on the components of the contact force and moment. Specifically, the frictional components of $\boldsymbol{\lambda}_i$ are constrained to lie inside a limit surface, denoted by $\mathcal{L}_i$, that scales linearly with the product $\mu_i f_{in}$, where μ_i is the coefficient of friction at contact i. In the case of Coulomb friction, the limit surface is a circle of radius $\mu_i f_{in}$. The Coulomb friction cone $\mathcal{F}_i$ is a subset of $\mathbb{R}^3$

$$\mathcal{F}_i = \{(f_{in}, f_{it}, f_{io}) | \sqrt{f_{it}^2 + f_{io}^2} \leq \mu_i f_{in}\} \,. \tag{38.57}$$

More generally, the friction laws of interest have limit surfaces defined in the space of friction components, $\mathbb{R}^{n_{\lambda i}-1}$ and friction cones $\mathcal{F}_i$ defined in the space of $\boldsymbol{\lambda}_i$, $\mathbb{R}^{n_{\lambda i}}$. They can be written as follows

$$\mathcal{F}_i = \{\boldsymbol{\lambda}_i \in \mathbb{R}^{n_{\lambda i}} \mid ||\boldsymbol{\lambda}_i||_w \leq f_{in}\} \,, \tag{38.58}$$

where $||\boldsymbol{\lambda}_i||_w$ denotes a weighted quadratic norm of the friction components at contact i. The limit surface is defined by $||\boldsymbol{\lambda}_i||_w = f_{in}$.

Table 38.10 defines useful weighted quadratic norms for the three contact models: PwoF, HF, and SF. The parameter μ_i is the friction coefficient for the tangential forces, υ_i is the torsional friction coefficient, and a is the characteristic length of the object that is used to ensure consistent units for the norm of the SF model.

Table 38.10 Norms for the three main contact models

Model	$\|\|\boldsymbol{\lambda}_i\|\|_w$
PwoF	0
HF	$\frac{1}{\mu_i}\sqrt{f_{it}^2 + f_{io}^2}$
SF	$\frac{1}{\mu_i}\sqrt{f_{it}^2 + f_{io}^2} + \frac{1}{a\,\upsilon_i}\lvert m_{in}\rvert$

Remark 38.4

There are several noteworthy points to be made about the friction cones. First, all of them implicitly or explicitly constrain the normal component of the contact force to be non-negative. The cone for SF contacts has a cylindrical limit surface with circular cross section in the (f_{it}, f_{io})-plane and rectangular cross section in the (f_{it}, m_{in})-plane. With this model, the amount of torsional friction that can be transmitted is independent of the lateral friction load. An improved model that couples the torsional friction limit with the tangential limit was studied by *Howe* and *Cutkosky* [38.46].

A Force Closure Definition

One common definition of force closure can be stated simply by modifying condition (38.45) to allow each contact force to lie in its friction cone rather than along the contact normal. Because this definition does not consider the hand's ability to control contact forces, this definition will be referred to as *frictional form closure*. A grasp will be said to have frictional form closure if and only if the following condition is satisfied

$$\left.\begin{array}{l}\mathbf{G}\boldsymbol{\lambda} = -\boldsymbol{g} \\ \boldsymbol{\lambda} \in \mathcal{F}\end{array}\right\} \forall\, \boldsymbol{g} \in \mathbb{R}^{n_v}\,,$$

where $\mathcal{F}$ is the composite friction cone defined as:

$$\begin{aligned}\mathcal{F} &= \mathcal{F}_1 \times \cdots \times \mathcal{F}_{n_c} \\ &= \{\boldsymbol{\lambda} \in \mathbb{R}^m | \boldsymbol{\lambda}_i \in \mathcal{F}_i;\ i = 1, \ldots, n_c\}\,,\end{aligned}$$

and each $\mathcal{F}_i$ is defined by (38.58) and one of the models listed in Table 38.10.

Letting $\mathrm{Int}(\mathcal{F})$ denote the interior of the composite friction cone, *Murray* et al. give the following equivalent definition [38.19]:

Definition 38.9

(Proposition 5.2, Murray et al.) A grasp has *frictional form closure* if and only if the following conditions are satisfied:

1. $\mathrm{rank}(\mathbf{G}) = n_v$
2. $\exists\, \boldsymbol{\lambda}$ such that $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$ and $\boldsymbol{\lambda} \in \mathrm{Int}(\mathcal{F})$.

These conditions define what Murray et al. call *force closure*. The force closure definition adopted here is stricter than frictional form closure; it additionally requires that the hand be able to control the internal object forces.

Definition 38.10

A grasp has *force closure* if and only if

$$\mathrm{rank}(\mathbf{G}) = n_v,\ \mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}\,,$$

and there exists $\boldsymbol{\lambda}$ such that $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$ and $\boldsymbol{\lambda} \in \mathrm{Int}(\mathcal{F})$.

The full row rank condition on the matrix $\mathbf{G}$ is the same condition required for form closure, although $\mathbf{G}$ is different from $\mathbf{G}_n$ used to determine form closure. If the rank test passes, then one must still find $\boldsymbol{\lambda}$ satisfying the remaining three conditions. Of these, the null space intersection test can be performed easily by linear programming techniques, but the friction cone constraint is quadratic, and thus forces one to use non-linear programming techniques. While exact non-linear tests have been developed [38.47], only approximate tests will be presented here.

Approximate Force Closure Tests

Any of the friction cones discussed can be approximated as the non-negative span of a finite number n_g of generators $\boldsymbol{s}_{ij}$ of the friction cone. Given this, one can represent the set of applicable contact wrenches at contact i as follows

$$\mathbf{G}_i\boldsymbol{\lambda}_i = \mathbf{S}_i\boldsymbol{\sigma}_i, \quad \boldsymbol{\sigma}_i \geq 0\,,$$

where $\mathbf{S}_i = [\boldsymbol{s}_{i1} \cdots \boldsymbol{s}_{in_g}]$ and $\boldsymbol{\sigma}_i$ is a vector of non-negative generator weights. If contact i is frictionless, then $n_g = 1$ and $\mathbf{S}_i = [\hat{\boldsymbol{n}}_i^{\mathrm{T}}((\boldsymbol{c}_i - \boldsymbol{p}) \times \hat{\boldsymbol{n}}_i)^{\mathrm{T}}]^{\mathrm{T}}$.

If contact i is of type HF, we represent the friction cone by the non-negative sum of uniformly spaced contact force generators (Fig. 38.11) whose non-negative span approximates the Coulomb cone with an inscribed regular polyhedral cone. This leads to the following def-

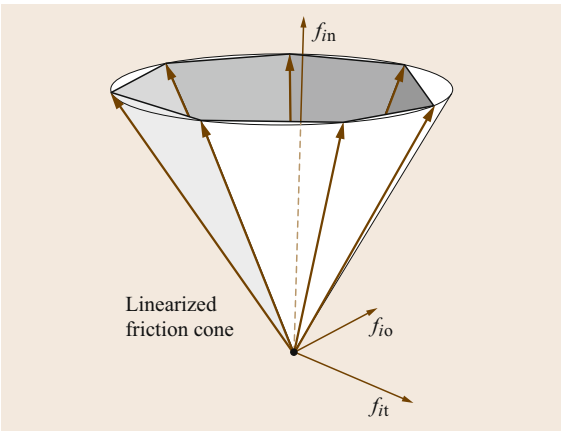


Fig. 38.11 Quadratic cone approximated as a polyhedral cone with seven generators

inition of $\mathbf{S}_i$

$$\mathbf{S}_i = \begin{pmatrix} \cdots & 1 & \cdots \\ \cdots & \mu_i \cos(2k\pi/n_g) & \cdots \\ \cdots & \mu_i \sin(2k\pi/n_g) & \cdots \end{pmatrix}, \tag{38.59}$$

where the index k varies from one to n_g. If one prefers to approximate the quadratic friction cone by a circumscribing polyhedral cone, one simply replaces μ_i in the above definition with $\mu_i/\cos(\pi/n_g)$.

The adjustment needed for the SF model is quite simple. Since the torsional friction in this model is decoupled from the tangential friction, its generators are given by $[1\ 0\ 0\ \pm b\upsilon_i]^T$. Thus $\mathbf{S}_i$ for the SF model is

$$\mathbf{S}_i = \left(\begin{array}{ccc|cc} \cdots & 1 & \cdots & 1 & 1 \\ \cdots & \mu_i \cos(2k\pi/n_g) & \cdots & 0 & 0 \\ \cdots & \mu_i \sin(2k\pi/n_g) & \cdots & 0 & 0 \\ \hline \cdots & 0 & \cdots & b\upsilon_i & -b\upsilon_i \end{array}\right), \tag{38.60}$$

where b is the characteristic length used to unify units. The set of total contact wrenches that may be applied by the hand without violating the contact friction law at any contact can be written as

$$\mathbf{G}\boldsymbol{\lambda} = \mathbf{S}\boldsymbol{\sigma}, \quad \boldsymbol{\sigma} \geq \mathbf{0},$$

where

$$\mathbf{S} = (\mathbf{S}_1, \cdots, \mathbf{S}_{n_g})$$

and

$$\boldsymbol{\sigma} = (\boldsymbol{\sigma}_1^T \cdots \boldsymbol{\sigma}_{n_g}^T)^T.$$

It is convenient to reformulate the friction constraints in a dual form

$$\mathbf{F}_i \boldsymbol{\lambda}_i \geq \mathbf{0}. \tag{38.61}$$

In this form, each row of $\mathbf{F}_i$ is normal to a face formed by two adjacent generators of the approximate cone. For an HF contact, row i of $\mathbf{F}_i$ can be computed as the cross product of $\boldsymbol{s}_i$ and $\boldsymbol{s}_{i+1}$. In the case of an SF contact, the generators are of dimension four, so simple cross products will not suffice. However, general methods exist to perform the conversion from the generator form to the face normal form [38.39].

The face normal constraints for all contacts can be combined into the following compact form

$$\mathbf{F}\boldsymbol{\lambda} \geq \mathbf{0}, \tag{38.62}$$

where $\mathbf{F} = \text{Blockdiag}(\mathbf{F}_1, \ldots, \mathbf{F}_{n_c})$.

Let $\boldsymbol{e}_i \in \mathbb{R}^{n_{\lambda i}}$ be the first row of $\mathbf{H}_i$. Further let

$$\boldsymbol{e} = [\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{n_c}] \in \mathbb{R}^{n_\lambda}$$

and let

$$\mathbf{E} = \text{Blockdiag}(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{n_c}) \in \mathbb{R}^{n_\lambda \times n_c}.$$

The following linear program is a quantitative test for frictional form closure. The optimal objective function value d^* is a measure of the distance the contact forces are from the boundaries of their friction cones, and hence a crude measure of how far a grasp is from losing frictional form closure.

LP2: maximize: d
subject to: $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{F}\boldsymbol{\lambda} - \mathbf{1}d \geq \mathbf{0}$
$d \geq 0$
$\boldsymbol{e}\boldsymbol{\lambda} \leq n_c$.

The last inequality in LP2 is simply the sum of the magnitudes of the normal components of the contact forces. After solving LP2, if $d^* = 0$ frictional form closure does not exist, but if $d^* > 0$, then it does.

If the grasp has frictional form closure, the last step to determine the existence of force closure is to verify the condition $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^T) = \mathbf{0}$. If it holds, then the grasp has force closure. This condition is easy to verify with another linear program LP3.

LP3: maximize: d
subject to: $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{J}^T\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{E}\boldsymbol{\lambda} - \mathbf{1}d \geq \mathbf{0}$
$d \geq 0$
$\boldsymbol{e}\boldsymbol{\lambda} \leq n_c$.

In summary, force closure testing is a three-step process.

Approximate Force Closure Test.

1. Compute rank($\mathbf{G}$).
   a) If rank($\mathbf{G}$) $\neq n_\upsilon$, then force closure does not exist. Stop.
   b) If rank($\mathbf{G}$) $= n_\upsilon$, continue.
2. Solve LP2: *Test frictional form closure.*
   a) If $d^* = 0$, then frictional form closure does not exist. Stop.
   b) If $d^* > 0$, then frictional form closure exists and d^* is a crude measure of how far the grasp is from losing frictional form closure.

3. Solve LP3: *Test control of internal force.*
   a) If $d^* > 0$, then force closure does not exist.
   b) If $d^* = 0$, then force closure exists.

Variation of the Test. A variation of the approximate force closure test arises when the object is partially constrained with the bilateral constraints described by (38.53), in such a cases the definitions of frictional form closure becomes

$$\begin{aligned} \overline{\mathbf{G}}\boldsymbol{\lambda} &= \mathbf{0}\,, \\ \boldsymbol{\lambda} &\in \operatorname{int}(\mathcal{F})\,, \end{aligned} \tag{38.63}$$

where $\overline{\mathbf{G}} = \mathbf{A}_B\mathbf{G}$, $\mathbf{A}_B$ is an annihilator of the column space of $\mathbf{B}$. Similarly, the force closure condition can be stated as

$$\begin{aligned} \overline{\mathbf{G}}\boldsymbol{\lambda} &= \mathbf{0}\,, \\ \boldsymbol{\lambda} &\in \operatorname{int}(\mathcal{F})\,, \\ \mathcal{N}(\overline{\mathbf{G}}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) &= \mathbf{0}\,, \end{aligned} \tag{38.64}$$

Since the frictional form closure definition (38.63) is analogous to Definition 38.9 and the force closure definition (38.64) is analogous to Definition 38.10, the force closure test can be applied by substituting $\overline{\mathbf{G}}$ for $\mathbf{G}$, provided that, $(n_v - \operatorname{rank}(\mathbf{B}))$ is substituted for n_v. For detailed information on the derivation of these conditions, please refer to [38.42].

See Example 1, Part 6.

Planar Simplifications

In planar grasping systems, the approximate method described above is exact. This is because the SF models are meaningless, since rotations about the contact normal would cause motions out of the plane. With regard to the HF model, for planar problems, the quadratic friction cone becomes linear, with its cone represented exactly as

$$\mathbf{F}_i = \frac{1}{\sqrt{1+\mu_i^2}} \begin{pmatrix} \mu_i & 1 \\ \mu_i & -1 \end{pmatrix}. \tag{38.65}$$

Nguyen's graphical form closure test can be applied to planar grasps with two frictional contacts [38.45]. The only change is that the four contact normals are replaced by the four generators of the two friction cones. However, the test can only determine frictional form closure, since it does not incorporate the additional information needed to determine force closure.

38.5 Examples

38.5.1 Example 1: Grasped Sphere

Part 1: $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{J}}$

Figure 38.12 shows a planar projection of a three-dimensional sphere of radius r grasped by two fingers, which make two contacts at angles θ_1 and θ_2. The frames $\{\boldsymbol{C}\}_1$ and $\{\boldsymbol{C}\}_2$ are oriented so that their $\hat{\boldsymbol{o}}$-directions point out of the plane of the figure (as indicated by the small filled circles at the contact points). The axes of the frames $\{\boldsymbol{N}\}$ and $\{\boldsymbol{B}\}$ were chosen to be axis-aligned with coincident origins located at the center of the sphere. The z-axes are pointing out of the page. Observe that since the two joint axes of the left finger are perpendicular to the (x, y)-plane, it operates in that plane for all time. The other finger has three revolute joints. Because its first and second axes, $\hat{z}_3$ and $\hat{z}_4$, currently lie in the plane, rotation about $\hat{z}_3$ will cause $\hat{z}_4$ to attain an out-of-plane component and would cause the finger tip at contact 2 to leave the plane.

In the current configuration, the rotation matrix for the i-th contact frame is defined as follows

$$\mathbf{R}_i = \begin{pmatrix} -\cos(\theta_i) & \sin(\theta_i) & 0 \\ -\sin(\theta_i) & -\cos(\theta_i) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{38.66}$$

The vector from the origin of $\{\boldsymbol{N}\}$ to the i-th contact point is given by

$$\boldsymbol{c}_i - \boldsymbol{p} = r \begin{pmatrix} \cos(\theta_i) & \sin(\theta_i) & 0 \end{pmatrix}^{\mathrm{T}}. \tag{38.67}$$

Substituting into (38.3) and (38.6) yields the complete grasp matrix for contact i

$$\tilde{\mathbf{G}}_i = \left(\begin{array}{ccc|ccc} -c_i & s_i & 0 & & & \\ -s_i & -c_i & 0 & & \mathbf{0} & \\ 0 & 0 & 1 & & & \\ \hline 0 & 0 & rs_i & -c_i & s_i & 0 \\ 0 & 0 & -rc_i & -s_i & -c_i & 0 \\ 0 & -r & 0 & 0 & 0 & 1 \end{array}\right), \tag{38.68}$$

where $\mathbf{0} \in \mathbb{R}^{3\times 3}$ is the zero matrix and c_i and s_i are abbreviations for $\cos(\theta_i)$ and $\sin(\theta_i)$, respectively. The complete grasp matrix is defined as: $\tilde{\mathbf{G}} = (\tilde{\mathbf{G}}_1\ \tilde{\mathbf{G}}_2) \in \mathbb{R}^{6\times 12}$.

The accuracy of this matrix can be verified by inspection, according to Remark 38.2. For example, the first column is the unit wrench of the unit contact normal; the first three components are the direction cosines

of $\hat{\boldsymbol{n}}_i$ and the last three are $(\boldsymbol{c}_i - \boldsymbol{p}) \times \hat{\boldsymbol{n}}_i$. Since $\hat{\boldsymbol{n}}_i$ is collinear with $(\boldsymbol{c}_i - \boldsymbol{p})$, the cross products (the last three components of the column) are zero. The last three components of the second column represent the moment of $\hat{\boldsymbol{t}}_i$ about the $\boldsymbol{x}$-, $\boldsymbol{y}$-, and $\boldsymbol{z}$-axes of $\{N\}$. Since $\hat{\boldsymbol{t}}_i$ lies in the (x, y)-plane, the moments with the x- and $\boldsymbol{y}$-axes are zero. Clearly $\hat{\boldsymbol{t}}_i$ produces a moment of $-r$ about the $\boldsymbol{z}$-axis.

Construction of the hand Jacobian $\tilde{\mathbf{J}}_i$ for contact i requires knowledge of the joint axis directions and the origins of the frames fixed to the links of each finger. Figure 38.13 shows the hand in the same configuration as in Fig. 38.12, but with some additional data needed to construct the hand Jacobian. Assume that the origins of the joint frames lie in the plane of the figure. In the current configuration, the quantities of interest for contact 1, expressed in $\{\boldsymbol{C}\}_1$ are

$$\boldsymbol{c}_1 - \boldsymbol{\zeta}_1 = \begin{pmatrix} l_2 & l_1 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.69}$$

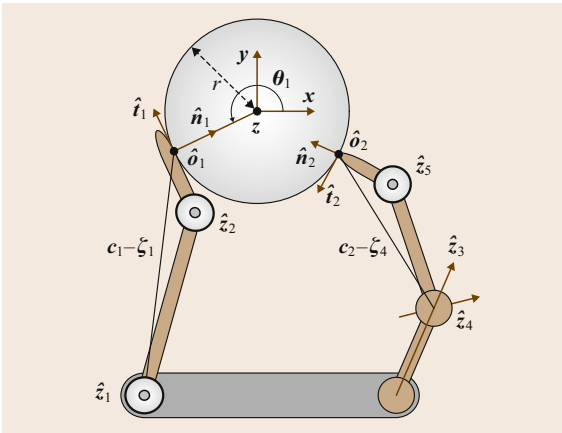


Fig. 38.12 A sphere grasped by a two-fingered hand with 5 revolute joints

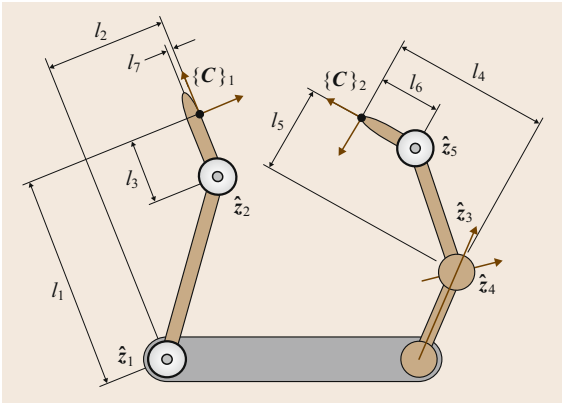


Fig. 38.13 Relevant data for the hand Jacobian in Fig. 38.12

$$\boldsymbol{c}_1 - \boldsymbol{\zeta}_2 = \begin{pmatrix} l_7 & l_3 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.70}$$

$$\hat{\boldsymbol{z}}_1 = \hat{\boldsymbol{z}}_2 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \tag{38.71}$$

The quantities of interest for contact 2, in $\{\boldsymbol{C}\}_2$ are

$$\boldsymbol{c}_2 - \boldsymbol{\zeta}_3 = \boldsymbol{c}_2 - \boldsymbol{\zeta}_4 = \begin{pmatrix} l_4 & l_5 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.72}$$

$$\boldsymbol{c}_2 - \boldsymbol{\zeta}_5 = \begin{pmatrix} l_6 & 0 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.73}$$

$$\hat{\boldsymbol{z}}_3 = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.74}$$

$$\hat{\boldsymbol{z}}_4(q_3) = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 & 1 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.75}$$

$$\hat{\boldsymbol{z}}_5(q_3, q_4) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \tag{38.76}$$

Generally all of the components of the $\boldsymbol{c} - \boldsymbol{\zeta}$ and $\hat{\boldsymbol{z}}$ vectors (including the components that are zero in the current configuration), are functions of $\boldsymbol{q}$ and $\boldsymbol{u}$. The dependencies of the $\hat{\boldsymbol{z}}$ vectors are shown explicitly.

Substituting into (38.14), (38.11) and (38.8) yields the complete hand Jacobian $\tilde{\mathbf{J}} \in \mathbb{R}^{12\times 5}$

$$\tilde{\mathbf{J}} = \left(\begin{array}{cc|ccc} -l_1 & -l_3 & & & \\ l_2 & l_7 & & & \\ 0 & 0 & & & \\ 0 & 0 & & \mathbf{0} & \\ 0 & 0 & & & \\ 1 & 1 & & & \\ \hline & & 0 & 0 & 0 \\ & & 0 & 0 & l_6 \\ & & l_4 & \frac{\sqrt{2}}{2}(l_4 + l_5) & 0 \\ & \mathbf{0} & 0 & -\frac{\sqrt{2}}{2} & 0 \\ & & -1 & -\frac{\sqrt{2}}{2} & 0 \\ & & 0 & 0 & 1 \end{array}\right) .$$

The horizontal dividing line partitions $\tilde{\mathbf{J}}$ into $\tilde{\mathbf{J}}_1$ (on top) and $\tilde{\mathbf{J}}_2$ (on the bottom). The columns correspond to joints 1 through 5. The block diagonal structure is a result of the fact that finger i directly affects only contact i.

Example 1, Part 2: G and J

Assume that the contacts in Fig. 38.12 are both of type SF. Then the selection matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \left(\begin{array}{cccccc|ccccc} 1 & 0 & 0 & 0 & 0 & 0 & & & & & \\ 0 & 1 & 0 & 0 & 0 & 0 & & & \mathbf{0} & & \\ 0 & 0 & 1 & 0 & 0 & 0 & & & & & \\ 0 & 0 & 0 & 1 & 0 & 0 & & & & & \\ \hline & & & & & & 1 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & 0 & 1 & 0 & 0 & 0 & 0 \\ & & & \mathbf{0} & & & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & & & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right) ,$$

thus the matrices $\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{8\times 6}$ and $\mathbf{J} \in \mathbb{R}^{8\times 5}$ are constructed by removing rows 5, 6, 11, and 12 from $\tilde{\mathbf{G}}^{\mathrm{T}}$ and $\tilde{\mathbf{J}}$

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{cccccc} -c_1 & -s_1 & 0 & 0 & 0 & 0 \\ s_1 & -c_1 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & r\,s_1 & -r\,c_1 & 0 \\ 0 & 0 & 0 & -c_1 & -s_1 & 0 \\ \hline -c_2 & -s_2 & 0 & 0 & 0 & 0 \\ s_2 & -c_2 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & r\,s_2 & -r\,c_2 & 0 \\ 0 & 0 & 0 & -c_2 & -s_2 & 0 \end{array}\right), \quad (38.77)$$

$$\mathbf{J} = \left(\begin{array}{cc|ccc} -l_1 & -l_3 & & & \\ l_2 & l_7 & & \mathbf{0} & \\ 0 & 0 & & & \\ 0 & 0 & & & \\ \hline & & 0 & 0 & 0 \\ & & 0 & 0 & l_6 \\ \mathbf{0} & & l_4 & \frac{\sqrt{2}}{2}(l_4+l_5) & 0 \\ & & 0 & -\frac{\sqrt{2}}{2} & 0 \end{array}\right). \quad (38.78)$$

Notice that changing the contact models is easily accomplished by removing more rows. Changing contact 1 to HF would eliminate the fourth rows from $\mathbf{G}^{\mathrm{T}}$ and $\mathbf{J}$. Changing it to PwoF would eliminate the second, third and fourth rows of $\mathbf{G}^{\mathrm{T}}$ and $\mathbf{J}$. Changing the model at contact 2 would remove either just the eighth row or the sixth, seventh, and eighth rows.

Example 1, Part 3: Reduce to Planar Case

The grasp shown in Fig. 38.12 can be reduced to a planar problem by following the explicit formulas given above, but it can also be done by understanding the physical interpretations of the various rows and columns of the matrices. Proceed by eliminating velocities and forces that are out of the plane. This can be done by removing the z-axes from $\{N\}$ and $\{B\}$, and the $\hat{\boldsymbol{o}}$-directions at the contacts. Further, joints 3 and 4 must be locked. The resulting $\mathbf{G}^{\mathrm{T}}$ and $\mathbf{J}$ are constructed eliminating certain rows and columns. $\mathbf{G}^{\mathrm{T}}$ is formed by removing rows 3, 4, 7, and 8 and columns 3, 4, and 5. $\mathbf{J}$ is formed by removing rows 3, 4, 7, and 8 and columns 3 and 4 yielding

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{ccc} -c_1 & -s_1 & 0 \\ s_1 & -c_1 & -r \\ \hline -c_2 & -s_2 & 0 \\ s_2 & -c_2 & -r \end{array}\right), \quad (38.79)$$

$$\mathbf{J} = \left(\begin{array}{cc|c} -l_1 & -l_3 & 0 \\ l_2 & l_7 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & l_6 \end{array}\right). \quad (38.80)$$

Example 1, Part 4: Grasp Classes

The first column of Table 38.11 reports the dimensions of the main subspaces of $\mathbf{J}$ and $\mathbf{G}$ for the sphere grasping example with different contact models. Only non-trivial null spaces are listed.

In the case of two HF contact models, all four null spaces are non-trivial, so the system satisfies the conditions for all four grasp classes. The system is graspable because there is an internal force along the line segment connecting the two contact points. Indeterminacy is manifested in the fact that the hand cannot resist a moment acting about that line. Redundancy is seen to exist since joint 3 can be used to move contact 2 out of the plane of the figure, but joint 4 can be rotated in the opposite direction to cancel this motion. Finally, the grasp is defective, because the contact forces and the instantaneous velocities along the $\hat{\boldsymbol{o}}_1$ and $\hat{\boldsymbol{n}}_2$ directions of contact 1 and 2, respectively, cannot be controlled through the joint torques and velocities. These interpretations are borne out in the null space basis matrices below, computed using $r = 1$, $\cos(\theta_1) = -0.8 = -\cos(\theta_2)$, $\sin(\theta_1) = \cos(\theta_2) = -0.6$, $l_1 = 3$, $l_2 = 2$, $l_3 = 1$, $l_4 = 2$, $l_5 = 1$, $l_6 = 1$, and $l_7 = 0$

$$\mathbf{N}(\mathbf{J}) \approx \begin{pmatrix} 0 \\ 0 \\ -0.73 \\ 0.69 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{G}^{\mathrm{T}}) \approx \begin{pmatrix} 0 \\ 0 \\ 0.51 \\ 0.86 \\ 0 \\ 0 \end{pmatrix}, \quad (38.81)$$

$$\mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ -0.42 \\ 0 \\ 0.57 \\ 0.42 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}. \quad (38.82)$$

Table 38.11 Dimensions of main subspaces and classifications of grasp studied in Example 1

Models	Dimension	Class
HF,HF	$\dim \mathcal{N}(\mathbf{J}) = 1$	Redundant
	$\dim \mathcal{N}(\mathbf{G}^{\mathrm{T}}) = 1$	Indeterminate
	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 2$	Defective
SF,HF	$\dim \mathcal{N}(\mathbf{J}) = 1$	Redundant
	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 3$	Defective
HF,SF	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 2$	Defective
SF,SF	$\dim \mathcal{N}(\mathbf{G}) = 2$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 3$	Defective

Notice that changing either contact to SF makes it possible for the hand to resist external moments applied about the line containing the contacts, so the grasp loses indeterminacy, but retains graspability (with squeezing still possible along the line of the contacts). However, if contact 2 is the SF contact, the grasp loses its redundancy. While the second contact point can still be moved out of the plane by joint 3 and back in by joint 4, this canceled translation of the contact point yields a net rotation about $\hat{\boldsymbol{n}}_2$ (this also implies that the hand can control the moment applied to the object along the line containing the contacts). Changing to SF at contact 2 does *not* affect the hand's inability to move contact 1 and contact 2 in the $\hat{\boldsymbol{o}}_1$ and $\hat{\boldsymbol{n}}_2$ directions, so the defectivity property is retained.

Example 1, Part 5: Desirable Properties

Assuming contact model types of SF and HF at contacts 1 and 2, respectively, $\mathbf{G}$ is full row rank and so $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = \mathbf{0}$ (see Table in part 4 of this example). Therefore, as long as the hand is sufficiently dexterous, it can apply any wrench in $\mathbb{R}^6$ to the object. Also, if the joints are locked, object motion will be prevented. Assuming the same problem values used in the previous part of this problem, the matrix $\mathbf{G}^{\mathrm{T}}$ is

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{cccccc} -c_1 & -s_1 & 0 & 0 & 0 & 0 \\ s_1 & -c_1 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & rs_1 & -rc_1 & 0 \\ 0 & 0 & 0 & -c_1 & -s_1 & 0 \\ \hline -c_2 & -s_2 & 0 & 0 & 0 & 0 \\ s_2 & -c_2 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & rs_2 & -rc_2 & 0 \end{array}\right) \tag{38.83}$$

Bases for the three non-trivial null spaces are

$$\mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \tag{38.84}$$

$$\mathbf{N}(\mathbf{J}) \approx \begin{pmatrix} 0 \\ 0 \\ -0.73 \\ 0.69 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ -0.42 \\ 0 \\ 0 \\ 0.57 \\ 0.42 \\ 0 \end{pmatrix}. \tag{38.85}$$

Since $\mathcal{R}(\mathbf{J})$ is four-dimensional and $\mathcal{N}(\mathbf{G})$ is one-dimensional, the maximum dimension of $\mathcal{R}(\mathbf{J}) + \mathcal{N}(\mathbf{G})$ cannot be more than 5, and therefore, the hand cannot control all possible object velocities. For example, the contact velocity $\boldsymbol{v}_{\mathrm{cc}} = (0\,0\,0\,0.8\,0\,0\,0)^{\mathrm{T}}$ is in $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$, and so cannot be controlled by the fingers. It is also equal to 0.6 times the third column of $\mathbf{G}^{\mathrm{T}}$ plus the 4th column of $\mathbf{G}^{\mathrm{T}}$ and therefore is in $\mathcal{R}(\mathbf{G}^{\mathrm{T}})$. Since the mapping between $\mathcal{R}(\mathbf{G})$ and $\mathcal{R}(\mathbf{G}^{\mathrm{T}})$ is one-to-one and onto, this uncontrollable contact velocity corresponds to a unique uncontrollable object velocity, $\boldsymbol{v} = (0\,0\,0.6\,1\,0\,0)$. In other words, the hand cannot cause the center of the sphere to translate in the z-direction, while also rotating about the $\boldsymbol{x}$-axis (and not other axes simultaneously).

On the question of controlling all internal object forces, the answer is *yes*, since $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) = 0$. This conclusion is clear from the fact that $\mathcal{N}(\mathbf{G})$ has non-zero values in the first, second, and sixth positions, while all columns of $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ have zeros in those positions.

Example 1, Part 6: Force Closure

Again assume that contacts 1 and 2 on the grasped sphere were modeled as SF and HF contacts, respectively. Under this assumption, $\mathbf{G}$ is full row rank, and the internal force corresponds to equal and opposite contact forces. For frictional form closure to exist, the internal force must lie within the friction cones. Choosing r and the sines and cosines of θ_1 and θ_2 as in example 1, part 4, frictional form closure can be shown to exist if both friction coefficients are greater than 0.75. For this grasp, since $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) = \mathbf{0}$, frictional form closure is equivalent to force closure.

The plot in Fig. 38.14 was generated by fixing μ_2 at a specific value and varying μ_1 from 0.5 to 2.0. Notice that for $\mu_1 < 0.75$, force closure does not exist regardless of the value of μ_2. The metric increases smoothly until a specific value of μ_1. From that point on, the friction coefficient at contact 2 is the limiting factor. To increase the metric further, μ_2 must be increased.

38.5.2 Example 2: Grasped Polygon in the Plane

Part 1: G and J

Figure 38.15 shows a planar hand grasping a polygon. Finger 1 (on the right) contains two joints numbered 1 and 2. Finger 2 contains joints 3–7, which are numbered in increasing order moving from the palm distally. The inertial frame has been chosen to lie inside the object, with its $\boldsymbol{x}$-axis passing through contacts 1 and 2, and collinear with the normal vector of contact 2.

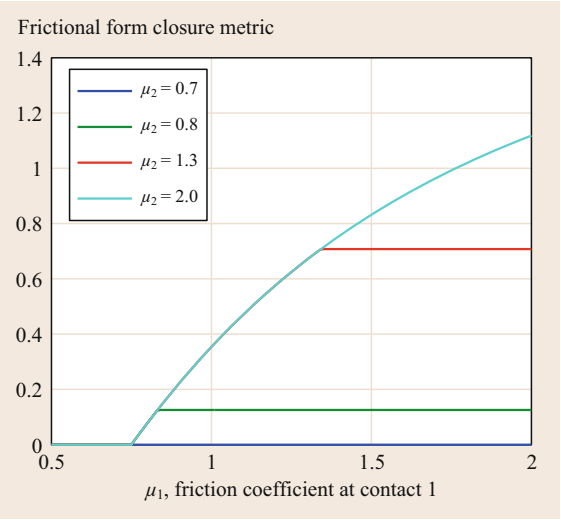


Fig. 38.14 Plot of force closure metric versus friction coefficient on contact 1

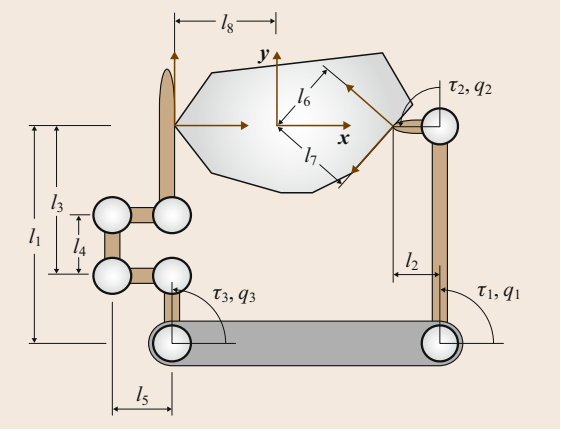


Fig. 38.15 Planar hand with 2 fingers and 7 joints grasping a polygonal object

The rotation matrices are given by

$$\mathbf{R}_1 = \begin{pmatrix} -0.8 & -0.6 \\ 0.6 & -0.8 \end{pmatrix}, \quad \mathbf{R}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{38.86}$$

Assuming HF contacts, $\mathbf{G}$ is given as follows

$$\mathbf{G} = \left(\begin{array}{cc|cc} -0.8 & -0.6 & 1 & 0 \\ 0.6 & -0.8 & 0 & 1 \\ l_6 & -l_7 & 0 & -l_8 \end{array}\right). \tag{38.87}$$

Notice that the first two columns of $\mathbf{G}$ correspond to the normal and tangential unit vectors at contact 1. The third and fourth columns correspond to contact 2.

Assuming HF contacts and all joints are active (i. e., not locked), $\mathbf{J}$ is

$$\mathbf{J}^{\mathrm{T}} = \left(\begin{array}{cc|cc} 0.8l_1 & 0.6l_1 & \multicolumn{2}{c}{\multirow{2}{*}{$\mathbf{0}$}} \\ -0.6l_2 & 0.8l_2 & & \\ \hline & & -l_1 & 0 \\ & & -l_3 & 0 \\ \multicolumn{2}{c|}{\mathbf{0}} & -l_3 & l_5 \\ & & -l_3 + l_4 & l_5 \\ & & -l_3 + l_4 & 0 \end{array}\right). \tag{38.88}$$

The first two columns of $\mathbf{J}^{\mathrm{T}}$ are the torques required to produce a unit force in the $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{t}}_1$ directions at contact 1. The horizontal line through the matrix partitions the contributions for the first finger (the upper part) and second finger. Notice that both $\mathbf{J}^{\mathrm{T}}$ and $\mathbf{G}$ are full column rank.

Example 2, Part 2: Grasp Classes

This example clearly illustrates the physical qualities of the various grasp classes without introducing features that can cloud the descriptions.

We now discuss the details of the four grasp classes using the previous planar example. During these discussions it is useful to choose non-dimensional values for the parameters in the grasping system. The lengths were assumed to have the following values (the results are the same, regardless of a particular choice of units, so units are not specified)

$$l_1 = 2.89\,, \quad l_2 = 0.75\,, \quad l_3 = 1.97\,, \tag{38.89}$$
$$l_4 = 0.80\,, \quad l_5 = 0.80\,, \quad l_6 = 0.90\,, \tag{38.90}$$
$$l_7 = 1.20\,, \quad l_8 = 1.35\,. \tag{38.91}$$

Redundant. Redundancy exists if $\mathcal{N}(\mathbf{J})$ is non-trivial. Assuming that both contacts are hard contacts and all the joints are active, $\mathrm{rank}(\mathbf{J}) = 4$, so $\mathcal{N}(\mathbf{J})$ is three-dimensional. A basis for $\mathcal{N}(\mathbf{J})$ was obtained using Matlab's `null()` function

$$\mathbf{N}(\mathbf{J}) \approx \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline -0.49 & -0.31 & -0.27 \\ 0.53 & 0.64 & -0.17 \\ 0.49 & -0.50 & -0.02 \\ -0.49 & 0.50 & 0.02 \\ -0.02 & 0.01 & 0.95 \end{array}\right). \tag{38.92}$$

Since the first two rows are zero, $\mathcal{N}(\mathbf{J})$ does not include motions of the first finger (on the right of the palm). To understand this, assume the object is fixed in the plane. Then the first finger cannot maintain sticking contact at contact 1 unless its joints are also fixed.

The three non-zero columns corresponding to finger 2 show that there are 3 basis motions of its joints that allow the finger contact to stick to the object contact. For example, the first column shows that if joint 3 moves roughly as much as joints 4, 5, and 6, but in the opposite direction as joints 4 and 5 and in the same directionas joint 6, while joint 7 is more or less fixed, then contact 2 will be maintained.

Notice that finger 2 contains a parallelogram. Because of this geometry, one can see that the vector $(0\,0\,0\;-1\,1\;-1\,1)^{\mathrm{T}}$ is an element of $\mathcal{N}(\mathbf{J})$. The velocity interpretation of this vector is that the link of the finger connected to the palm, and the link touching the object remain fixed in space, while the parallelogram moves as a simple four-bar mechanism. Similarly, joint actions in $\mathcal{N}(\mathbf{J})$ do not affect the contact forces, but cause internal hand velocities. Also, notice that since $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$, the entire space of possible generalized velocities and forces at the contacts can be generated by the joints.

Indeterminate. As noted above, with HF contact models, the system is graspable. However, replacing the HF models with PwoF models removes the tangent force components in the $\hat{\boldsymbol{t}}_1$ and $\hat{\boldsymbol{t}}_2$ directions. This effectively removes columns 2 and 4 from $\mathbf{G}$, which guarantees that the system will be indeterminate. The reduced matrix is denoted by $\mathbf{G}_{(1,3)}$. In this case $\mathcal{N}\left(\mathbf{G}^{\mathrm{T}}_{(1,3)}\right)$ is

$$\mathbf{N}\left(\mathbf{G}^{\mathrm{T}}_{(1,3)}\right) \approx \begin{pmatrix} 0 \\ -0.83 \\ 0.55 \end{pmatrix}. \tag{38.93}$$

Physically, this basis vector corresponds to moving the object such that the point coincident with the origin of $\{\boldsymbol{N}\}$ moves directly downward, while the object rotates counter clockwise. Also, if the analogous force and moment were applied to the object, the frictionless contacts could not maintain equilibrium.

Graspable. With two HF contact models in force, $\mathrm{rank}(\mathbf{G}) = 3$, so $\mathcal{N}(\mathbf{G})$ is one–dimensional and the system is graspable. The null space basis vector of the grasp matrix is

$$\mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ 0.42 \\ \hline 0.71 \\ 0 \end{pmatrix}. \tag{38.94}$$

The physical interpretation of this basis vector is two opposing forces acting through the two contact points. Recall that because the contact model is kinematic, there is no consideration of contact friction. However, given the direction of the contact normal relative to the line of the internal force, one can see that if the coefficient of friction is not greater than 0.75, squeezing tightly will cause sliding at contact 1, thus violating the kinematic contact model.

Defective. In a defective grasp, $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \neq \mathbf{0}$. Given that the original $\mathbf{J}$ is full row rank, the grasp is *not* defective. However, it can be made defective by locking a number of joints and/or changing the hand's configuration so that $\mathbf{J}$ is no longer full rank. For example, locking joints 4, 5, 6, and 7 makes finger 2 a single-link finger with only joint 3 active. In this new grasping system, $\mathbf{J}^{\mathrm{T}}_{(1,2,3)}$ is simply the first three rows of the original $\mathbf{J}^{\mathrm{T}}$ given in (38.88), where the subscript is the list of indices of active joints. The null space basis vector is

$$\mathbf{N}(\mathbf{J}^{\mathrm{T}}_{(1,2,3)}) = \begin{pmatrix} 0 \\ 0 \\ \hline 0 \\ 1 \end{pmatrix}. \tag{38.95}$$

This grasp is defective, since there is a subspace of contact velocities and forces that cannot be controlled by joint generalized velocities and forces. Since only the last component of $\mathbf{N}(\mathbf{J}^{\mathrm{T}}_{(1,2,3)})$ is non-zero, it would be impossible for the hand to give the contact point 2 on the object a velocity in the $\hat{\boldsymbol{t}}_2$-direction while maintaining the contact. This is also clear from the arrangement of joint 3, contact 2, and the direction of the contact normal. The dual interpretation is that forces in $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ are resisted by the structure and the corresponding joint loads is zero, or equivalently that those forces are not controllable by the hand. Notice that if the model of contact 2 were changed to point-without-friction, then $\mathbf{N}(\mathbf{J}^{\mathrm{T}}_{(1,2,3)}) = \mathbf{0}$ and the system would no longer be defective.

38.5.3 Example 3: Hyperstatic Grasps

Part 1: $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{J}}$

Figure 38.16 shows a planar projection of a three-dimensional sphere of radius l grasped by one finger only, with 3 revolute joints, through 3 contacts. The frames $\{\boldsymbol{C}\}_1$, $\{\boldsymbol{C}\}_2$ and $\{\boldsymbol{C}\}_3$ are oriented so that their $\hat{\boldsymbol{o}}$-directions point out of the plane of the figure (as indicated by the small filled circle). The axes of the frames $\{\boldsymbol{N}\}$ and $\{\boldsymbol{B}\}$ were chosen to be axis-aligned with coincident origins located at the center of the sphere. The z-axes are pointing out of the page. Observe that since

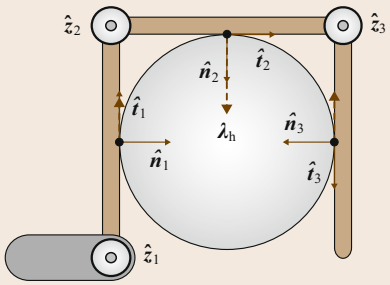


Fig. 38.16 A sphere grasped by a finger with three revolute joints. The force direction $\boldsymbol{\lambda}_h$ (*dashed line*) is a force that belongs to both $\mathcal{N}(\mathbf{G})$ and $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ and causes hyperstaticity

the three joint axes of the finger are perpendicular to the (x, y)-plane, the grasp operates in that plane for all time.

Assume that the width of all the links of the robotic hand is zero. Rotation matrices $\mathbf{R}_i$ and vectors $\boldsymbol{c}_i - \boldsymbol{p}$ for $i = 1, \ldots, 3$, can be computed as in (38.66) and (38.67) considering that $\theta_1 = \pi$, for contact 1, $\theta_2 = \pi/2$ and $\theta_3 = 0$, for contact 2 and 3, respectively. Finally, the complete grasp matrix is

$$\tilde{\mathbf{G}}^{\mathrm{T}} = \begin{pmatrix} \tilde{\mathbf{G}}_1 & \tilde{\mathbf{G}}_2 & \tilde{\mathbf{G}}_2 \end{pmatrix}^{\mathrm{T}} \in \mathbb{R}^{18\times 6} ,$$

where $\tilde{\mathbf{G}}_i$ is as defined in (38.68)

$$\tilde{\mathbf{G}}^{\mathrm{T}} = \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & l & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & l & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & -l & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

Construction of the complete hand Jacobian $\tilde{\mathbf{J}}_i$ for contact i requires knowledge of the joint axis directions and the origins of the frames fixed to the links of each finger. Assume that the origins of the Denavit–Hartenberg (DH) frames lie in the plane of the figure. In the current configuration, the quantities of interest for contact 1, expressed directly in $\{\boldsymbol{N}\}$ are

$$\begin{aligned} \boldsymbol{c}_1 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} 0 & l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \end{aligned}$$

The quantities of interest for contact 2, in $\{\boldsymbol{N}\}$ are

$$\begin{aligned} \boldsymbol{c}_2 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} l & 2l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \boldsymbol{c}_2 - \boldsymbol{\zeta}_2 &= \begin{pmatrix} l & 0 & 0 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_2 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \end{aligned}$$

The quantities of interest for contact 3, in $\{\boldsymbol{N}\}$ are

$$\begin{aligned} \boldsymbol{c}_3 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} 2l & l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \boldsymbol{c}_3 - \boldsymbol{\zeta}_2 &= \begin{pmatrix} 2l & -l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \boldsymbol{c}_3 - \boldsymbol{\zeta}_3 &= \begin{pmatrix} 0 & -l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_2 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_3 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \end{aligned}$$

The complete hand Jacobian $\tilde{\mathbf{J}} \in \mathbb{R}^{18\times 3}$ (contact velocities are expressed in $\{\boldsymbol{C}\}_i$) is

$$\tilde{\mathbf{J}} = \left(\begin{array}{ccc} -l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ \hline -l & -l & 0 \\ -2l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline l & -l & -l \\ -2l & -2l & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{array}\right) .$$

The horizontal dividing lines partition $\tilde{\mathbf{J}}$ into $\tilde{\mathbf{J}}_1$ (on top), $\tilde{\mathbf{J}}_2$, and $\tilde{\mathbf{J}}_3$ (on the bottom). The columns correspond to joints 1 through 3.

Example 3, Part 2: G and J

Assume that the 3 contacts in Fig. 38.16 are of type HF. Then the selection matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{pmatrix}, \tag{38.96}$$

where $\mathbf{I}$ and $\mathbf{0}$ are in $\mathbb{R}^{3\times3}$, thus matrices $\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{9\times6}$ and $\mathbf{J} \in \mathbb{R}^{9\times3}$ are obtained by removing rows related to rotations from $\tilde{\mathbf{G}}^{\mathrm{T}}$ and $\tilde{\mathbf{J}}$

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & l & 0 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & l & 0 & 0 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & -l & 0 \end{array}\right),$$

$$\mathbf{J} = \left(\begin{array}{ccc} l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline l & l & 0 \\ 2l & 0 & 0 \\ 0 & 0 & 0 \\ \hline l & l & l \\ 2l & 2l & 0 \\ 0 & 0 & 0 \end{array}\right).$$

Example 3, Part 3: Grasp Classes

The first column of Table 38.12 reports the dimensions of the main subspaces of $\mathbf{J}^{\mathrm{T}}$ and $\mathbf{G}$ for the sphere grasping example with three hard finger contacts. Only non-trivial null spaces are listed.

The system is defective because there are generalized contact forces belonging to the subspace that are resisted by the structure, which correspond to zero joint actions

$$\mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \left(\begin{array}{cccccc} 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

The first three columns represent generalized forces acting at the three contact points in a direction perpendicular to the plane of the Fig. 38.16. The fourth column corresponds to a contact force applied only along the $\hat{\boldsymbol{t}}_1$ direction.

System is graspable because the subspace of internal forces is three-dimensional; a possible basis matrix is

$$\mathbf{N}(\mathbf{G}) = \left(\begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ \hline 1 & 0 & 2 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{array}\right).$$

The three force vectors of subspace $\mathbf{N}(\mathbf{G})$ are easily identified from Fig. 38.16. Note that all forces are expressed in local contact frames. The first column vector of $\mathbf{N}(\mathbf{G})$ represents opposed forces at contacts 1 and 2 along the line joining contacts 1 and 2. The second column vector parameterizes opposed forces at contacts 1 and 3 along the line joining contacts 1 and 3. The last vector represents forces along direction $\boldsymbol{\lambda}_h$, shown as the (dashed lines) in Fig. 38.16. Note that this direction (in wrench intensity space) corresponds to two upward friction forces at the left and right contacts and one downward with double the magnitude from the center of the top link in the work space.

Finally, the grasp is hyperstatic because

$$\mathbf{N}(\mathbf{G}) \cap \mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \left(\begin{array}{c} 0 \\ 1 \\ 0 \\ \hline 2 \\ 0 \\ 0 \\ \hline 0 \\ -1 \\ 0 \end{array}\right) \neq 0 .$$

Table 38.12 Dimensions of main subspaces and classification of grasp given in Example 3

Dimension	Class
$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 6$	Defective
$\dim \mathcal{N}(\mathbf{G}) = 3$	Graspable
$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) = 1$	Hyperstatic

Hyperstatic forces in this subspace, are internal forces that cannot be controlled through the hand joints. In Fig. 38.16 the internal force $\boldsymbol{\lambda}_h$ that is also in $\mathbf{N}(\mathbf{J}^{\mathrm{T}})$ is reported.

The grasp in Fig. 38.16 is an example of *power grasp*, style of grasp mentioned earlier that uses many contact points not only on the fingertips but also on the links of fingers and the palm [38.8, 28, 33].

All power grasps are kinematically defective ($\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \neq \mathbf{0}$) and usually are hyperstatic. According to Sect. 38.2.2, rigid-body modeling is not sufficient to capture the overall system behavior, because the generalized contact forces in $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}})$ leave the dynamics indeterminate.

Many approaches have been used to overcome rigid-body limitation in hyperstatic grasps such as those proposed in [38.16, 20, 21] where visco–elastic contact models have been used to solve the force indeterminacy. In [38.48], authors found that a sufficient condition for hyperstaticity is $m > q + 6$ where m is the dimension of contact force vector.

38.5.4 Example 4: Duality

Consider a frictionless disc constrained to translate in the plane; (Fig. 38.17). In this problem $n_v = 2$, so the space of applied contact forces and object velocities is the plane $\mathbb{R}^2$. In the top pair of pictures, a single (fixed) contact point imposes a half space constraint on the instantaneous velocity and limits the force at a frictionless contact to the ray. Both the ray and the (dark gray) half space are defined by the contact normal pointing into the object. Notice that the ray and half space are dual cones. When two contacts are present, the (light gray) force cone becomes the non-negative span of the two contact normals and the velocity cone is its dual. With the addition of the third contact, the grasp has form closure as indicated by the degeneration of the velocity cone to the origin and the expansion of the force cone to become equal to the plane.

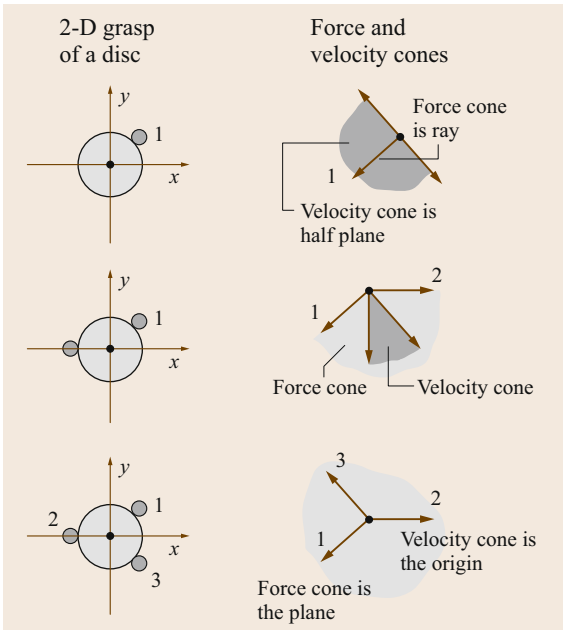


Fig. 38.17 Case of a translating disc in the plane: Relationship between frictionless contacts and possible disc velocities and net contact forces

It is important to point out that the discussion of the dual cones applies to three-dimensional bodies after replacing the contact normals with the columns of $\mathbf{G}$.

38.5.5 Example 5: Form Closure

Part 1: Unilateral Constraints

Form closure of a spatial object requires seven unilateral contacts, which is difficult to illustrate. Therefore, the only form closure example analyzed in this chapter is the planar problem shown in Fig. 38.18. In the plane only four unilateral contacts are needed. In this problem, even though the fourth contact is at a vertex of the object, the contact normal is still well-defined. The angle α of the finger is allowed to vary, and it can be shown that form closure exists if α lies in the interval: $1.0518 < \alpha < \frac{\pi}{2}$. Notice that a critical value of α occurs when the lower edge of C_2 contains contact point 3 ($\alpha \approx 1.0518$) and contact point 2 ($\alpha = \frac{\pi}{2}$). Beyond these angles, the cone C_1 and C_2 can no longer *see* each other.

Choosing the frame for analysis with origin at the fourth contact point, the grasp matrix for this example

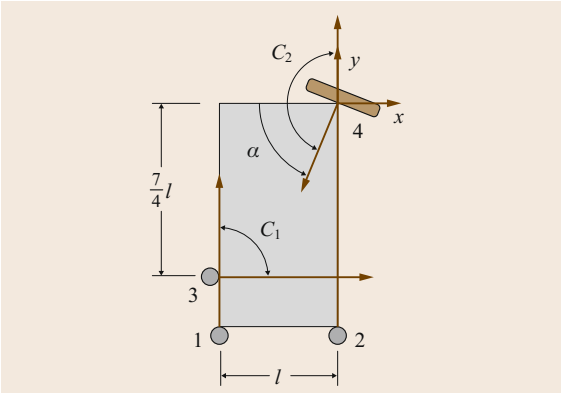


Fig. 38.18 Plot of closure metrics versus angle of contact for $1.04 < \alpha < 1.59$

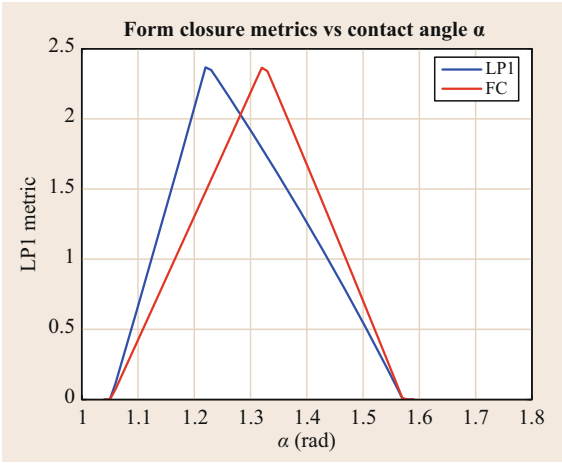


Fig. 38.19 Planar grasp with first-order form closure if $1.052 < \alpha < \frac{\pi}{2}$

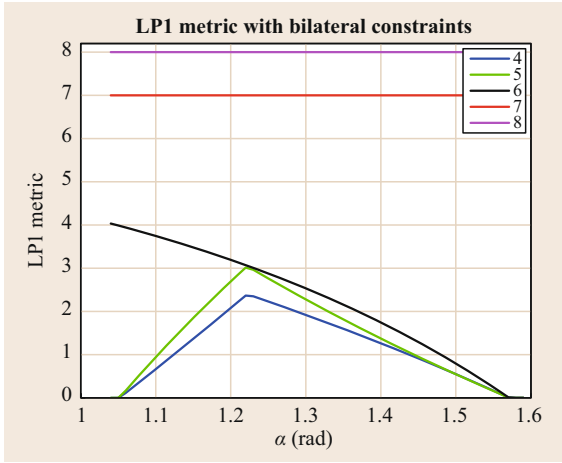


Fig. 38.20 First-order form closure metric for Fig. 38.18 with progressively more contacts converted to bilateral

is

$$\mathbf{G} = \begin{pmatrix} 0 & 0 & 1 & -\cos(\alpha) \\ 1 & 1 & 0 & -\sin(\alpha) \\ -l & 0 & \frac{7}{4}l & 0 \end{pmatrix} . \tag{38.97}$$

Form closure was tested for a range of angles. The blue curve in Fig. 38.19 is the LP1 metric ($d^* * n_c$ returned by the function `formClosure.m`, available from [38.43]), which indicates that the grasp farthest from losing form closure has $\alpha \approx 1.222$ radians, which is the configuration shown in Fig. 38.18. For comparison purposes, the Ferrari–Canny metric (scaled to have the same maximum as the LP1 metric) is plotted in red. They agree on the angles for which form closure exists, but differ on the optimal α.

Example 5, Part 2: Bilateral Constraints

Let us next consider form closure with a mix of bilateral and unilateral constraints using (38.54). For example, if contact 1 is treated as a bilateral constraint and the remaining are considered to be unilateral, $\mathbf{G}_n$ and $\mathbf{B}$ are

$$\mathbf{B} = \begin{pmatrix} 0 \\ 1 \\ -l \end{pmatrix} , \quad \mathbf{G}_n = \begin{pmatrix} 0 & 1 & -\cos(\alpha) \\ 1 & 0 & -\sin(\alpha) \\ 0 & \frac{7}{4}l & 0 \end{pmatrix} . \tag{38.98}$$

Additional contacts can be converted to bilateral by moving their corresponding columns from $\mathbf{G}_n$ to $\mathbf{B}$. Figure 38.20 shows five plots of the form closure metric LP1. In the legend, *4* refers to the grasp with all four contacts treated as unilateral, *5* means that contact 1 has been converted to bilateral as defined by the previous equation, *6* means that contacts 1 and 2 have been converted to bilateral, etc. The plots show an important property of the LP1 metric; the maximum value is equal to the number of unilateral contacts (bilateral contacts count as two unilateral contacts), which is attained for curves *7* and *8*. For this particular problem, these maxima are achieved for all α.

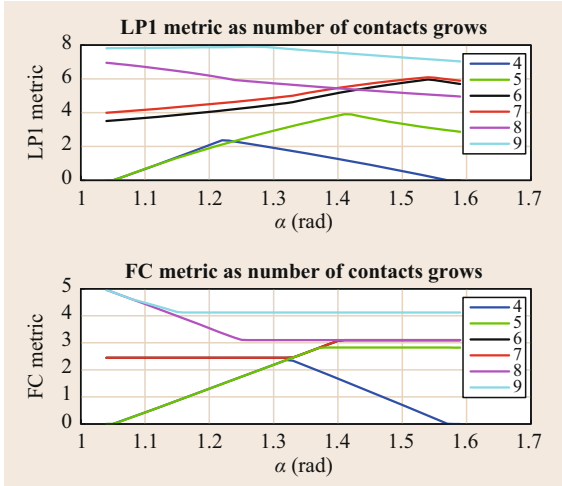


Fig. 38.21 Comparison of the monotonicity properties of the LP1 and Ferrari–Canny form closure metrics

Example 5, Part 3: Monotonicity

If one were to zoom in on curves *4* and *5* near $\alpha = 1.5$, one would see that LP1 is not monotonic as constraints are added. That is, when contact *1* is converted from unilateral (curve *4*) to bilateral (curve *5*), for some values of α, the LP1 metric is smaller despite the additional constraint. To demonstrate monotonicity, the monotonic Ferrari–Canny metric and the non-monotonic LP1

metric were computed as a function of α for a sequence of grasps. Starting with the original four contacts, five unilateral contacts were added: $\{(0, -1.3l), (-0.5l, 0), (0, 1.75l), (-l, -0.6l), (-0.7l, 0)\}$. Figure 38.21 shows the metrics with 4, 5, 6, 7, 8, and 9 contacts, as indicated by the legend.

The LP1 metric does not increase monotonically as contacts are added for every α. Near $\alpha = 1.2$ adding the fifth contact reduces the metric slightly. Also, near $\alpha = 1.55$, adding the eighth contact reduces the metric. By contrast, the Ferrari–Canny metric is monotonic, although not strictly monotonic; the plots labeled *6* and *7* are identical for all α and there are other intervals of α where the metric is constant as contacts are added. For example, near $\alpha = 1.35$, the grasps with 5, 6, and 7 contacts all have the same metric value.

38.6 Conclusion and Further Reading

A great deal of understanding of grasping systems can be derived from the simple linear kinematic, dynamic, and contact models presented in this chapter. The most widely used grasp classifications and closure properties can all be derived from these models under the rigid-body assumption. Linearizing these models leads to metrics and tests that can be computed efficiently using computational linear algebra and linear programming techniques. Grasp synthesis tools built on these tests take object and hand models as inputs and return a set of possible grasp configurations as outputs (see for example [38.3]). In-depth discussions of grasp kinematics and grasp classifications can be found in [38.15, 20, 20, 34, 48–52].

One has to wonder what insights have been lost as a result of the simplifying assumptions made in this chapter. For the interested reader, there are a host of papers that analyze grasping systems under more sophisticated assumptions. In general, bodies are curved and compliant [38.17, 28, 35, 53–55] and contact friction models are not quite as simple as the linearized ones so widely adopted. For example, if a contact has to resist a moment about its normal, its effective tangential friction coefficient is reduced [38.46, 56]. In this chapter, the quadratic Coulomb friction cone was approximated by a polyhedral cone. The analysis problems are more difficult when using the quadratic cone, but they are quite tractable [38.47, 57].

In principle, a properly designed grasping system could be controlled to maintain all contacts, but worldly realities can lead to unwanted slipping or twisting. This leads us back to the topic of grasp stability, which is too often equated to grasp closure. However, grasp closure is really equivalent to the existence of equilibrium, which is a necessary, but not sufficient, condition for stability. The common definition of stability outside of the field of robotic grasping requires that when a system is deflected from an equilibrium point, the system returns to this point. From this perspective, and under the assumption of no slipping at the contact points, it is known that all closure grasps are stable, but the converse is not necessarily true [38.53, 54, 58, 59]. However, stability analysis when contacts slide is still an open question.

Given a stable grasp, another important consideration not discussed in this chapter is that of grasp force distribution problem, i. e., finding *good* actuator torques and contact forces to balance a given external load applied to the object. This problem was studied in the context of walking machines first by *McGhee* and *Orin* [38.60] and later by several others [38.61, 62]. *Kumar* and *Waldron* applied similar techniques to force distribution problems in grasping [38.63]. Work by *Han* et al. and *Buss* et al. solved the force distribution problem with non-linear friction cone constraints by taking advantage of convex optimization techniques [38.47, 57, 64]. In power grasps, this problem of finding a good distribution is difficult, because the space of controllable contact wrenches is severely restricted by the large number of contacts [38.21, 32, 33, 65].

Grasp synthesis largely depends on the structure of the robotic hands that are often very complex systems with many degrees of freedom, sensors and actuators, which are necessary to adapt to many different objects and tasks. An important research area, not discussed in this chapter, is that of designing simplified hands coupling some of the degrees of freedom, reducing the number of effective inputs, and leading to more efficient, simpler and reliable designs [38.23, 66]. A reduction of independent inputs is observed also in human hand movement data, where few variables, defined as postural synergies, explain most of the variance in the configurations of the hand while grasping different objects [38.25, 67]. The reduction of the number of independently controlled inputs in the hand affects grasp properties, and in particular the ability of the hand to dexterously controlling grasp forces and in-hand manipulation as discussed in [38.68–71]

All of the above considerations implicitly assume that a grasp has been achieved, which is no easy task. The bulk of today's research in robotic grasping can be fairly characterized as focused on *grasp acquisition*.

In other words, the problem is to move the hand from a state of no contact with the object to one in which a satisfactory grasp has been achieved. When a robot identifies an object to grasp, its knowledge of the object's pose and geometry are not perfect. Even if they were, the robot's control system could not move the hand perfectly to the desired grasping points. The hand will bump the object accidentally, altering its pose, possibly leading to grasp failure.

Areas of current research cover methods that exploit some detailed aspects of planned or sensed contact interactions that occur before achieving the final grasp and those that try to be robust to pre-grasp contact interactions. In the first category are quasistatic push-grasping, dynamic grasp acquisition, and perception. Push-grasping seeks out contact prior to wrapping the fingers in order to allow the object to settle into a good position against the palm. It has, so far, been applied to objects that can slide stably across a horizontal surface cluttered with object that are not to be grasped [38.72]. Dynamic planning combines a dynamic model of grasping that includes intermittent contact to design optimal controllers and grasping actions simultaneously [38.73]. The performance of both of the above in real-time implementations could be improved by new grasp perception methods, which estimate the pose of the object relative to the hand [38.74–76].

Impedance controllers are being developed to reduce the negative effects of unexpected contacts between the hand and object as the grasp is being formed [38.77] and between the object and environment during object transport tasks [38.78]. Independent contact regions are surface patches on the object which have the property: if a contact point is stabilized anywhere inside each region, then the grasp will have force closure [38.79, 80]. With this approach, a small amount of jostling will not cause grasp failure. Perhaps caging takes the independent regions idea to the extreme. Here the goal is to find a configuration of the hand that loosely surrounds the object with the additional condition that the object cannot escape without deforming itself or the hand [38.81, 82]. The challenges in caging are in finding a pre-cage configuration and a finger motion plan that impose the least restrictive accuracy requirements on the robots perception and control systems.

Video-References

VIDEO 551 Grasp analysis using the MATLAB toolbox SynGrasp
available from http://handbookofrobotics.org/view-chapter/04/videodetails/551

References

38.1 A. Bicchi: Hands for dextrous manipulation and powerful grasping: A difficult road towards simplicity, IEEE Trans. Robotics Autom. **16**, 652–662 (2000)

38.2 J.K. Salisbury: Kinematic and Force Analysis of Articulated Hands, Ph.D. Thesis (Stanford University, Stanford 1982)

38.3 A.T. Miller, P.K. Allen: GraspIt! A versatile simulator for robotic grasping, IEEE Robotics Autom. Mag. **11**(4), 110–122 (2004)

38.4 M. Malvezzi, G. Gioioso, G. Salvietti, D. Prattichizzo: SynGrasp: A matlab toolbox for underactuated and compliant hands, IEEE Robotics Autom. Mag. **22**(4), 52–68 (2015)

38.5 M. Malvezzi, G. Gioioso, G. Salvietti, D. Prattichizzo, A. Bicchi: Syngrasp: A matlab toolbox for grasp analysis of human and robotic hands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1088–1093

38.6 SynGrasp: A MATLAB Toolbox for Grasp Analysis of Human and Robotic Hands, http://syngrasp.dii.unisi.it/

38.7 M. Grebenstein, A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Hoppner, S. Jorg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimbock, S. Wolf, T. Wusthoff, G. Hirzinger: The DLR hand arm system, IEEE Conf. Robotics Autom. (2011) pp. 3175–3182

38.8 K. Salisbury, W. Townsend, B. Ebrman, D. DiPietro: Preliminary design of a whole-arm manipulation system (WAMS), Proc. IEEE Int. Conf. Robotics Autom. (1988) pp. 254–260

38.9 M.S. Ohwovoriole, B. Roth: An extension of screw theory, J. Mech. Des. **103**, 725–735 (1981)

38.10 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Oxford Univ. Press, Oxford 1978)

38.11 T.R. Kane, D.A. Levinson, P.W. Likins: *Spacecraft Dynamics* (McGraw Hill, New York 1980)

38.12 J.J. Craig: *Introduction to Robotics: Mechanics and Control*, 2nd edn. (Addison-Wesley, Reading 1989)

38.13 J.K. Salisbury, B. Roth: Kinematic and force analysis of articulated mechanical hands, J. Mech. Transm. Autom. Des. **105**(1), 35–41 (1983)

38.14 M.T. Mason, J.K. Salisbury Jr: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
38.15 A. Bicchi: On the closure properties of robotic grasping, Int. J. Robotics Res. **14**(4), 319–334 (1995)
38.16 D. Prattichizzo, A. Bicchi: Consistent task specification for manipulation systems with general kinematics, ASME J. Dyn. Syst. Meas. Control **119**(4), 760–767 (1997)
38.17 J. Kerr, B. Roth: Analysis of multifingered hands, Int. J. Robotics Res. **4**(4), 3–17 (1986)
38.18 G. Strang: *Introduction to Linear Algebra* (Wellesley-Cambridge Press, Wellesley 1993)
38.19 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robot Manipulation* (CRC, Boca Raton 1993)
38.20 D. Prattichizzo, A. Bicchi: Dynamic analysis of mobility and graspability of general manipulation systems, IEEE Trans. Robotics Autom. **14**(2), 241–258 (1998)
38.21 A. Bicchi: On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation, Int. J. Robotics Auton. Syst. **13**, 127–147 (1994)
38.22 A.M. Dollar, R.D. Howe: Joint coupling and actuation design of underactuated hands for unstructured environments, Int. J. Robotics Res. **30**, 1157–1169 (2011)
38.23 L. Birglen, T. Lalibertè, C. Gosselin: *Underactuated robotic hands*, Springer Tracts in Advanced Robotics (Springer, Berlin, Heidelberg 2008)
38.24 M.G. Catalano, G. Grioli, A. Serio, E. Farnioli, C. Piazza, A. Bicchi: Adaptive synergies for a humanoid robot hand, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012) pp. 7–14
38.25 M. Gabiccini, A. Bicchi, D. Prattichizzo, M. Malvezzi: On the role of hand synergies in the optimal choice of grasping forces, Auton. Robots **31**, 235–252 (2011)
38.26 M. Malvezzi, D. Prattichizzo: Evaluation of grasp stiffness in underactuated compliant hands, Proc. IEEE Int. Conf. Robotics Autom. (2013) pp. 2074–2079
38.27 S.F. Chen, I. Kao: Conservative congruence transformation for joint and cartesian stiffness matrices of robotic hands and fingers, Int. J. Robotics Res. **19**(9), 835–847 (2000)
38.28 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robotic hand, IEEE Trans. Robotics Autom. **5**(2), 151–165 (1989)
38.29 A. Albu-Schaffer, O. Eiberger, M. Grebenstein, S. Haddadin, C. Ott, T. Wimbock, S. Wolf, G. Hirzinger: Soft robotics, IEEE Robotics Autom, Mag. **15**(3), 20–30 (2008)
38.30 A. Bicchi: Force distribution in multiple whole-limb manipulation, Proc. IEEE Int. Conf. Robotics Autom. (1993)
38.31 F. Reuleaux: *The Kinematics of Machinery* (Macmillan, New York 1876), Republished by Dover, New York, 1963
38.32 T. Omata, K. Nagata: Rigid body analysis of the indeterminate grasp force in power grasps, IEEE Trans. Robotics Autom. **16**(1), 46–54 (2000)
38.33 J.C. Trinkle: The Mechanics and Planning of Enveloping Grasps, Ph.D. Thesis (University of Pennsylvania, Department of Systems Engineering, 1987)
38.34 K. Lakshminarayana: *Mechanics of Form Closure*, Tech. Rep., Vol. 78-DET-32 (ASME, New York 1978)
38.35 E. Rimon, J. Burdick: Mobility of bodies in contact i: A 2nd order mobility index for multiple-finger grasps, IEEE Trans. Robotics Autom. **14**(5), 696–708 (1998)
38.36 B. Mishra, J.T. Schwartz, M. Sharir: On the existence and synthesis of multifinger positive grips, Algorithmica **2**(4), 541–558 (1987)
38.37 P. Somov: Über Schraubengeschwindigkeiten eines festen Körpers bei verschiedener Zahl von Stützflächen, Z. Math. Phys. **42**, 133–153 (1897)
38.38 P. Somov: Über Schraubengeschwindigkeiten eines festen Körpers bei verschiedener Zahl von Stützflächen, Z. Math. Phys. **42**, 161–182 (1897)
38.39 A.J. Goldman, A.W. Tucker: Polyhedral convex cones. In: *Linear Inequalities and Related Systems*, ed. by H.W. Kuhn, A.W. Tucker (Princeton Univ., York 1956) pp. 19–40
38.40 X. Markenscoff, L. Ni, C.H. Papadimitriou: The geometry of grasping, Int. J. Robotics Res. **9**(1), 61–74 (1990)
38.41 D.G. Luenberger: *Linear and Nonlinear Programming*, 2nd edn. (Addison-Wesley, Reading 1984)
38.42 G. Muscio. J.C. Trinkle: *Grasp Closure Analysis of Bilaterally Constrained Objects*, Tech. Rep. Ser., Vol. 13-01 (Department of Computer Science, Rensselear Polytechnic Institute, Troy 2013)
38.43 Rensselaer Computer Science: http://www.cs.rpi.edu/twiki/bin/view/RoboticsWeb/LabSoftware
38.44 C. Ferrari, J. Canny: Planning optimal grasps, Proc. IEEE Int. Conf. Robotics Autom. (1986) pp. 2290–2295
38.45 V.D. Nguyen: *The Synthesis of Force Closure Grasps in the Plane, M.S. Thesis Ser.* (MIT Department of Mechanical Engineering, Cambridge 1985), AI-TR861
38.46 R.D. Howe, M.R. Cutkosky: Practical force-motion models for sliding manipulation, Int. J. Robotics Res. **15**(6), 557–572 (1996)
38.47 L. Han, J.C. Trinkle, Z. Li: Grasp analysis as linear matrix inequality problems, IEEE Trans. Robotics Autom. **16**(6), 663–674 (2000)
38.48 J.C. Trinkle: On the stability and instantaneous velocity of grasped frictionless objects, IEEE Trans. Robotics Autom. **8**(5), 560–572 (1992)
38.49 K.H. Hunt, A.E. Samuel, P.R. McAree: Special configurations of multi-finger multi-freedom grippers – A kinematic study, Int. J. Robotics Res. **10**(2), 123–134 (1991)
38.50 D.J. Montana: The kinematics of multi-fingered manipulation, IEEE Trans. Robotics Autom. **11**(4), 491–503 (1995)
38.51 Y. Nakamura, K. Nagai, T. Yoshikawa: Dynamics and stability in coordination of multiple robotic mechanisms, Int. J. Robotics Res. **8**, 44–61 (1989)
38.52 J.S. Pang, J.C. Trinkle: Stability characterizations of rigid body contact problems with coulomb friction, Z. Angew. Math. Mech. **80**(10), 643–663 (2000)

Avidbots Ex1016
Page 1019 of 2154

38.53 M.R. Cutkosky: *Robotic Grasping and Fine Manipulation* (Kluwer, Norwell 1985)
38.54 W.S. Howard, V. Kumar: On the stability of grasped objects, IEEE Trans. Robotics Autom. **12**(6), 904–917 (1996)
38.55 A.B.A. Cole, J.E. Hauser, S.S. Sastry: Kinematics and control of multifingered hands with rolling contacts, IEEE Trans. Autom. Control **34**, 398–404 (1989)
38.56 R.I. Leine, C. Glocker: A set-valued force law for spatial Coulomb–Contensou friction, Eur. J. Mech. A **22**(2), 193–216 (2003)
38.57 M. Buss, H. Hashimoto, J. Moore: Dexterous hand grasping force optimization, IEEE Trans. Robotics Autom. **12**(3), 406–418 (1996)
38.58 V. Nguyen: Constructing force-closure grasps, Int. J. Robotics Res. **7**(3), 3–16 (1988)
38.59 E. Rimon, J.W. Burdick: Mobility of bodies in contact II: How forces are generated by curvature effects, Proc. IEEE Int. Conf. Robotics Autom. (1998) pp. 2336–2341
38.60 R.B. McGhee, D.E. Orin: A mathematical programming approach to control of positions and torques in legged locomotion systems, Proc. ROMANCY (1976)
38.61 K. Waldron: Force and motion management in legged locomotion, IEEE J. Robotics Autom. **2**(4), 214–220 (1986)
38.62 T. Yoshikawa, K. Nagai: Manipulating and grasping forces in manipulation by multi-fingered grippers, Proc. IEEE Int. Conf. Robotics Autom. (1987) pp. 1998–2007
38.63 V. Kumar, K. Waldron: Force distribution in closed kinematic chains, IEEE J. Robotics Autom. **4**(6), 657–664 (1988)
38.64 M. Buss, L. Faybusovich, J. Moore: Dikin-type algortihms for dexterous grasping force optimization, Int. J. Robotics Res. **17**(8), 831–839 (1998)
38.65 D. Prattichizzo, J.K. Salisbury, A. Bicchi: Contact and grasp robustness measures: Analysis and experiments. In: *Experimental Robotics-IV, Lecture Notes in Control and Information Sciences*, Vol. 223, ed. by O. Khatib, K. Salisbury (Springer, Berlin, Heidelberg 1997) pp. 83–90
38.66 A.M. Dollar, R.D. Howe: The highly adaptive sdm hand: Design and performance evaluation, Int. J. Robotics Res. **29**(5), 585–597 (2010)
38.67 M.G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, A. Bicchi: Adaptive synergies for the design and control of the Pisa/IIT SoftHand, Int. J. Robotics Res. **33**(5), 768–782 (2014)
38.68 M.T. Ciocarlie, P.K. Allen: Hand posture subspaces for dexterous robotic grasping, Int. J. Robotics Res. **28**(7), 851–867 (2009)
38.69 T. Wimbock, B. Jahn, G. Hirzinger: Synergy level impedance control for multifingered hands, IEEE/RSJ Int Conf Intell. Robots Syst. (IROS) (2011) pp. 973–979
38.70 D. Prattichizzo, M. Malvezzi, M. Gabiccini, A. Bicchi: On the manipulability ellipsoids of underactuated robotic hands with compliance, Robotics Auton. Syst. **60**(3), 337–346 (2012)
38.71 D. Prattichizzo, M. Malvezzi, M. Gabiccini, A. Bicchi: On motion and force controllability of precision grasps with hands actuated by soft synergies, IEEE Trans. Robotics **29**(6), 1440–1456 (2013)
38.72 M.R. Dogar, S.S. Srinivasa: A framework for push-grasping in clutter, Robotics Sci. Syst. (2011)
38.73 M. Posa, R. Tedrake: Direct trajectory optimization of rigid body dynamical systems through contact, Proc. Workshop Algorithm. Found. Robotics (2012)
38.74 L. Zhang, J.C. Trinkle: The application of particle filtering to grasp acquistion with visual occlusion and tactile sensing, Proc. IEEE Int. Conf Robotics Autom. (2012)
38.75 P. Hebert, N. Hudson, J. Ma, J. Burdick: Fusion of stereo vision, force-torque, and joint sensors for estimation of in-hand object location, Proc. IEEE Int. Conf. Robotics Autom. (2011) pp. 5935–5941
38.76 S. Haidacher, G. Hirzinger: Estimating finger contact location and object pose from contact measurements in 3nd grasping, Proc. IEEE Int. Conf. Robotics Autom. (2003) pp. 1805–1810
38.77 T. Schlegl, M. Buss, T. Omata, G. Schmidt: Fast dextrous re-grasping with optimal contact forces and contact sensor-based impedance control, Proc. IEEE Int. Conf. Robotics Autom. (2001) pp. 103–108
38.78 G. Muscio, F. Pierri, J.C. Trinkle: A hand/arm controller that simultaneously regulates internal grasp forces and the impedance of contacts with the environment, IEEE Conf. Robotics Autom. (2014)
38.79 M.A. Roa, R. Suarez: Computation of independent contact regions for grasping 3-D objects, IEEE Trans. Robotics **25**(4), 839–850 (2009)
38.80 M.A. Roa, R. Suarez: Influence of contact types and uncertainties in the computation of independent contact regions, Proc. IEEE Int. Conf. Robotics Autom. (2011) pp. 3317–3323
38.81 A. Rodriguez, M.T. Mason, S. Ferry: From caging to grasping, Int. J. Robotics Res. **31**(7), 886–900 (2012)
38.82 C. Davidson, A. Blake: Error-tolerant visual planning of planar grasp, 6th Int. Conf. Comput. Vis. (1998) pp. 911–916

Avidbots Ex1016
Page 1020 of 2154

Multimedia Contents



39. Cooperative Manipulation

Fabrizio Caccavale, Masaru Uchiyama

This chapter is devoted to cooperative manipulation of a common object by means of two or more robotic arms. The chapter opens with a historical overview of the research on cooperative manipulation, ranging from early 1970s to very recent years. Kinematics and dynamics of robotic arms cooperatively manipulating a tightly grasped rigid object are presented in depth. As for the kinematics and statics, the chosen approach is based on the so-called symmetric formulation; fundamentals of dynamics and reduced-order models for closed kinematic chains are discussed as well. A few special topics, such as the definition of geometrically meaningful cooperative task space variables, the problem of load distribution, and the definition of manipulability ellipsoids, are included to give the reader a complete picture of modeling and evaluation methodologies for cooperative manipulators. Then, the chapter presents the main strategies for controlling both the motion of the cooperative system and the interaction forces between the manipulators and the grasped object; in detail, fundamentals of hybrid force/position control, proportional–derivative (PD)-type force/position control schemes, feedback linearization techniques, and impedance control approaches are given. In the last section further reading on advanced topics related to control of cooperative robots is suggested; in detail, advanced nonlinear control strategies are briefly discussed (i. e., intelligent control approaches, synchronization control, decentralized control); also, fundamental results on modeling and control of cooperative systems possessing some degree of flexibility are briefly outlined.

39.1 **Historical Overview** 990
39.2 **Kinematics and Statics** 991
39.2.1 Symmetric Formulation 992
39.2.2 Multifingered Manipulation 994
39.3 **Cooperative Task Space** 995
39.4 **Dynamics and Load Distribution** 996
39.4.1 Reduced-Order Models 997
39.4.2 Load Distribution 997
39.5 **Task-Space Analysis** 998
39.6 **Control** 999
39.6.1 Hybrid Control 999
39.6.2 PD Force/Motion Control 1000
39.6.3 Feedback Linearization Approaches 1001
39.6.4 Impedance Control 1002
39.7 **Conclusions and Further Reading** 1003
Video-References 1004
References 1004

It was not long after the emergence of robotics technologies that multi-arm robot systems began to be investigated by robotics researchers. In the early 1970s, research on this topic had already started. This interest was mainly due to typical limitations in applications of single-arm robots. It has been recognized, in fact, that many tasks that are difficult or impossible to execute by a single robot become feasible when two or more manipulators are employed in a cooperative way. Such tasks include, for instance, carrying heavy or large payloads, the assembly of multiple parts without using special fixtures, and handling of objects that are flexible or possess extra degrees of freedom. Research on this subject has been aimed at solving existing problems and opening up a new stream of applications in flexible manufacturing systems as well as in poorly structured environments (e.g., outer space and undersea).

39.1 Historical Overview

Examples of research work in the early days include that by *Fujii* and *Kurono* [39.1], *Nakano* et al. [39.2], and *Takase* et al. [39.3]. Already in those pieces of work important key issues in the control of multi-arm robots were investigated: master/slave control, force/compliance control, and task-space control. In [39.1] *Fujii* and *Kurono* proposed to adopt a compliance control concept for the coordination of multiple manipulators; they defined a task vector with respect to the object frame and controlled the compliance expressed in the same coordinate frame. An interesting feature of the work by *Fujii* and *Kurono* [39.1] and by *Takase* et al. [39.3] is that force/compliance control was implemented by exploiting back-drivability of the actuators, without using force/torque sensors. The importance of this technique in practical applications, however, was not recognized at that time, and more complex approaches using force/torque sensors lured people in robotics. *Nakano* et al. [39.2, 4] proposed a master/slave force control approach for the coordination of two arms carrying an object cooperatively and pointed out the necessity of force control for cooperative multiple robots.

In the 1980s, based on several fundamental theoretical results for single-arm robots, strong research on multi-arm robotic systems was renewed [39.5]. Definition of task vectors with respect to the object to be handled [39.6], dynamics and control of the closed kinematic chain formed by the multi-arm robot and the object [39.7, 8], and force control issues, such as hybrid position/force control [39.9–12], were explored. Through this research work, a strong theoretical background for the control of multi-arm robots has been formed, providing the basis for research on more advanced topics from the 1990s to today.

How to parameterize the constraint forces/moments on the object based on the dynamic model of the whole cooperative system has been recognized as a critical issue; in fact, this parametrization leads to the definition of task variables for the control and hence to an answer to one of the most frequently asked questions in the field of multi-arm robotics: how to control simultaneously the trajectory of the object, the mechanical stresses (internal forces/moments) acting on the object, load sharing among the arms, and even the external forces/moments on the object. Force decomposition may be a key to solving these problems and has been studied by *Uchiyama* and *Dauchez* [39.11, 12] and *Walker* et al. [39.13] as well as *Bonitz* and *Hsia* [39.14]. In detail, devising a geometrically clear parametrization of the internal forces/moments acting on the object has been recognized as an important problem, solved in [39.15, 16]. Several cooperative control schemes based on the sought parameterizations have been designed, including control of motion and force [39.11, 12, 17–19] and impedance/compliance control [39.20–22]. Other approaches include adaptive control [39.23, 24], kinematic control [39.25], task-space regulation [39.26], joint-space control [39.27, 28], and coordinated control [39.29].

Also, the definition of user-oriented task-space variables for coordinated control [39.26, 30] and the development of meaningful performance measures [39.31–34] have been fruitfully investigated in 1990s.

Load sharing among the arms is also an interesting issue on which many papers have been published [39.35–40]. Load sharing may be exploited both for optimal load distribution among arms and for robust holding of the object, when the object is held by the arms without being grasped rigidly. In both cases, anyhow, this becomes a problem of optimization and can be solved by either heuristic [39.41] or mathematical methods [39.42].

Other research efforts have been focused on cooperative handling of multibodied or even flexible objects [39.43–45]. Control of multi-flexible-arm robots has been investigated [39.46–48], since the merits of flexible-arm robots can be exploited in cooperative systems [39.49], i. e., lightweight structure, intrinsic compliance and hence safety, etc.

Robust holding of an object in the presence of slippage of end-effectors on the object may be achieved as well, if the slippage is detected correctly [39.50].

A more recent control framework for cooperative systems is so-called synchronization control [39.51, 52]; in this class of approaches the control problem is formulated in terms of suitably defined errors accounting for the motion synchronization between the manipulators involved in the cooperative task. As for the nonlinear control of cooperative manipulation systems, efforts have been spent on intelligent control (e.g., [39.53–55]) as well as on the investigation of control strategies in the presence of partial state feedback [39.56]. The problem of controlling underactuated cooperative manipulators is tackled, e.g., in [39.57].

The problem of the implementation of cooperative control strategies on conventional industrial robots has attracted the increasing interest of the research community. In fact, the control units of industrial robots do not present all the features needed to implement nonlinear torque control schemes, while the integration of force/torque sensing in standard industrial robot control units is often cumbersome and tends to be avoided in in-

dustry for many reasons: unreliability, cost, etc. Hence, the rebirth of the early methods, where force sensors were not used (*Fujii* and *Kurono* [39.1], *Inoue* [39.58]), has become attractive for industrial settings. Hybrid position/force control without using force/torque sensors has been successfully implemented [39.59]. Interesting results on the implementation of effective cooperative control strategies for industrial robots are presented, e.g., in [39.60], where a design approach that makes use of tool-based coordinate systems, trajectory generation, and distributed control of multiple robots is presented. Also, a task planning approach for multi-arm industrial robotic work-cells has been recently proposed in [39.30], together with a language for cooperative task programming.

Another interesting aspect, related to the reliability and safety of cooperative manipulation systems, is investigated in [39.61], where the use of nonrigid grippers is considered to avoid large internal forces and, at the same time, to achieve safe manipulation of the object, even in the presence of failures and unpredicted contacts with the external environment.

It is worth mentioning the strict relationship between issues related to grasping of objects by fingers/hands (widely described in Chap. 38) and those related to cooperative manipulation. In fact, in both cases, multiple manipulation structures grasp a commonly manipulated object. In multifingered hands only some motion components are transmitted through the contact point to the manipulated object (unilateral constraints), while cooperative manipulation via robotic arms is achieved by rigid (or near-rigid) grasp points and interaction takes place by transmitting all the motion components through the grasping points (bilateral constraints). While many common problems between the two fields can be tackled in a conceptually similar way (e.g., kinetostatic modeling, force control), many other are specific of each of the two application fields (e.g., form and force closure for multifingered hands). Interestingly, a common frame for both cooperative and multifingered manipulation has been proposed [39.25]; namely rolling/sliding of the grasp points on the object is taken into account by modeling the contacts via rotational/prismatic joints; then, the desired manipulator/finger joints trajectories are derived from the desired object motion by using numerical inverse kinematics algorithms. In [39.62–64], contributions to modeling and control of cooperative manipulation systems in the presence of different types of manipulators/object contact can be found. Moreover, since cooperative manipulators can be often viewed as closed-chain mechanisms, a strong link with research related to parallel manipulators (Chap. 18) exists, especially from a modeling point of view.

Finally, it is worth mentioning the important issue of cooperative transportation and manipulation of objects via multiple mobile robotic system [39.65, 66], which is currently subject of investigation.

39.2 Kinematics and Statics

Consider a system composed by M manipulators, each equipped with N_i joints ($i = 1, \ldots, M$). Let $\boldsymbol{p}_i$ denote the (3×1) vector of the position of the i-th end-effector coordinate frame, $\mathcal{T}_i$, with respect to a common base frame, $\mathcal{T}$; let $\mathbf{R}_i$ denote the (3×3) matrix which expresses the orientation of $\mathcal{T}_i$ with respect to the base frame $\mathcal{T}$.

Both $\boldsymbol{p}_i$ and $\mathbf{R}_i$ can be expressed as a function of the $(N_i \times 1)$ vector of the joints variables of each manipulator $\boldsymbol{q}_i$ through the direct kinematics equations

$$\begin{cases} \boldsymbol{p}_i = \boldsymbol{p}_i(\boldsymbol{q}_i) \,, \\ \mathbf{R}_i = \mathbf{R}_i(\boldsymbol{q}_i) \,. \end{cases} \tag{39.1}$$

Of course, the orientation of the end-effector frame may be expresses in terms of a minimal set of angles, i. e., a set of three Euler angles $\boldsymbol{\phi}_i$. In this case, direct kinematics can be expressed in terms of an operational space (pseudo-)vector $\boldsymbol{x}_i$ as follows

$$\boldsymbol{x}_i = \boldsymbol{k}_i(\boldsymbol{q}_i) = \begin{pmatrix} \boldsymbol{p}_i(\boldsymbol{q}_i) \\ \boldsymbol{\phi}_i(\boldsymbol{q}_i) \end{pmatrix} \,. \tag{39.2}$$

The linear $\dot{\boldsymbol{p}}_i$ and angular $\boldsymbol{\omega}_i$ velocity of the i-th end-effector can be collected in the (6×1) generalized velocities vector $\boldsymbol{v}_i = \left(\dot{\boldsymbol{p}}_i^{\mathrm{T}} \; \boldsymbol{\omega}_i^{\mathrm{T}}\right)^{\mathrm{T}}$. Then, the differential direct kinematics can be expressed as

$$\boldsymbol{v}_i = \mathbf{J}_i(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i \,, \tag{39.3}$$

where the $(6 \times N_i)$ matrix $\mathbf{J}_i$ is the so-called *geometric Jacobian* of the i-th manipulator (Chap. 2). When the velocity is expressed as the derivative of the operational space vector, differential kinematics takes a formally similar form

$$\dot{\boldsymbol{x}}_i = \frac{\partial \boldsymbol{k}_i(\boldsymbol{q}_i)}{\partial \boldsymbol{q}_i}\dot{\boldsymbol{q}}_i = \mathbf{J}_{\mathrm{A}i}(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i \,, \tag{39.4}$$

where the $(6 \times N_i)$ matrix $\mathbf{J}_{\mathrm{A}i}$ is the so-called *analytical Jacobian* of the i-th manipulator (Chap. 2).

Let us consider the (6×1) vector of the generalized forces acting at the i-th end-effector

$$\boldsymbol{h}_i = \begin{pmatrix} \boldsymbol{f}_i \\ \boldsymbol{n}_i \end{pmatrix} \,, \tag{39.5}$$

where $\boldsymbol{f}_i$ and $\boldsymbol{n}_i$ denote the force and moment, respectively. By invoking the principle of virtual work, a relation dual to (39.3) can be derived

$$\boldsymbol{\tau}_i = \mathbf{J}_i^{\mathrm{T}}(\boldsymbol{q}_i)\boldsymbol{h}_i \,, \tag{39.6}$$

where $\boldsymbol{\tau}_i$ is the $(N_i \times 1)$ vector of the forces/torques acting at the joints of the i-th manipulator.

For the sake of simplicity, consider a system of two cooperative robots (Fig. 39.1) manipulating a common object. Let C be a fixed point of the object (e.g., the center of mass of the object), whose position in the base frame is given by the vector $\boldsymbol{p}_{\mathrm{C}}$; moreover, let $\mathcal{T}_{\mathrm{C}}$ be a coordinate frame attached to object with origin in C. The *virtual stick* [39.11, 12] is defined as the vector $\boldsymbol{r}_i$ $(i = 1, 2)$ which determines the position of $\mathcal{T}_{\mathrm{C}}$ with respect to $\mathcal{T}_i$ $(i = 1, 2)$; it is assumed that $\boldsymbol{r}_i$ behaves as a rigid stick fixed to the i-th end-effector. Hence, each virtual stick, expressed in the frame $\mathcal{T}_i$ (or $\mathcal{T}_{\mathrm{C}}$), is a constant vector if the commonly grasped object is rigid and tightly grasped by each manipulator. In this case, the direct kinematics of each manipulator can be expressed in terms of a virtual end-effector frame $\mathcal{T}_{\mathrm{S},i} = \mathcal{T}_{\mathrm{C}}$, having the same orientation of $\mathcal{T}_{\mathrm{C}}$ and origin located in $\boldsymbol{p}_{\mathrm{S},i} = \boldsymbol{p}_i + \boldsymbol{r}_i = \boldsymbol{p}_{\mathrm{C}}$. Hence, the position and orientation at the tip of each virtual stick are given by $(i = 1, 2)$

$$\boldsymbol{p}_{\mathrm{S},i} = \boldsymbol{p}_{\mathrm{C}} \,, \quad \mathbf{R}_{\mathrm{S},i} = \mathbf{R}_{\mathrm{C}} \,.$$

The set of Euler angles corresponding to $\mathbf{R}_{\mathrm{S},i}$ will be denoted by the (3×1) vector $\boldsymbol{\phi}_{\mathrm{S},i}$. Hereafter, it is assumed that the object grasped by the two manipulators can be considered rigid (or nearly rigid) and tightly (or nearly tightly) attached to each end-effector. Hence, the displacements between the above defined frames can be considered null or negligible. Otherwise, if the manipulated object is subject to deformation (e.g., in the case of a flexible object) and/or the grasp is not tight (e.g., in the case of compliant grippers) the above defined coordinate frames, will be subject to nonnegligible displacements with respect to each other.

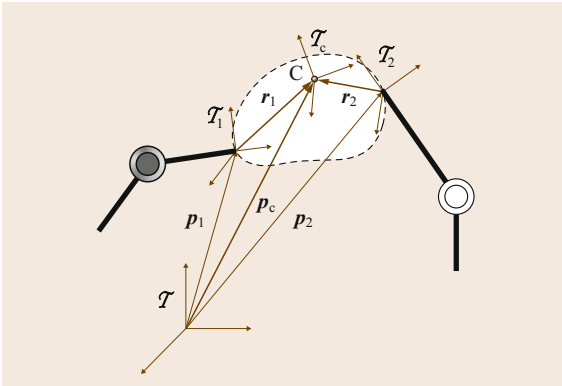


Fig. 39.1 Grasp geometry for a two-manipulator cooperative system manipulating a common object

Let $\boldsymbol{h}_{\mathrm{S},i}$ denote the vector of generalized forces acting at the tip of the i-th virtual stick; it can be easily verified that the following equation holds

$$\boldsymbol{h}_{\mathrm{S},i} = \begin{pmatrix} \mathbf{I}_3 & \mathbf{O}_3 \\ -\mathbf{S}(\boldsymbol{r}_i) & \mathbf{I}_3 \end{pmatrix} \boldsymbol{h}_i = \mathbf{W}_i \boldsymbol{h}_i \,, \tag{39.7}$$

where $\mathbf{O}_l$ and $\mathbf{I}_l$ denote, respectively, the null matrix and the identity matrix of $(l \times l)$ dimensions, and $\mathbf{S}(\boldsymbol{r}_i)$ is the (3×3) skew-symmetric matrix operator performing the cross product. It is worth noticing that $\mathbf{W}_i$ is always full rank.

By invoking the virtual works principle, a relation dual to (39.7) can easily be derived

$$\boldsymbol{v}_i = \begin{pmatrix} \mathbf{I}_3 & \mathbf{S}(\boldsymbol{r}_i) \\ \mathbf{O}_3 & \mathbf{I}_3 \end{pmatrix} \boldsymbol{v}_{\mathrm{S},i} = \mathbf{W}_i^{\mathrm{T}} \boldsymbol{v}_{\mathrm{S},i} \,, \tag{39.8}$$

where $\boldsymbol{v}_{\mathrm{S},i}$ is the generalized velocity vector of the virtual stick endpoint. When $\boldsymbol{r}_i = \mathbf{0}$, it is $\mathbf{W}_i = \mathbf{I}_6$; in other words, if the end-effector kinematics of each manipulator is referred to the corresponding virtual stick (or the object reduces to point), the forces and velocities at the two end-effectors coincide with their counterparts referred at the virtual sticks.

39.2.1 Symmetric Formulation

The kinetostatic formulation proposed by *Uchiyama* and *Dauchez* [39.12], i. e., the so-called *symmetric formulation*, is based on kinematic and static relationships between generalized forces/velocities acting at the object and their counterparts acting at the manipulators end-effectors (or at the virtual sticks endpoints).

Let us define first the *external forces* as the (6×1) vector of generalized forces given by

$$\boldsymbol{h}_{\mathrm{E}} = \boldsymbol{h}_{\mathrm{S},1} + \boldsymbol{h}_{\mathrm{S},2} = \mathbf{W}_{\mathrm{S}} \boldsymbol{h}_{\mathrm{S}} \,, \tag{39.9}$$

with $\mathbf{W}_{\mathrm{S}} = (\mathbf{I}_6 \; \mathbf{I}_6)$ and $\boldsymbol{h}_{\mathrm{S}} = \left(\boldsymbol{h}_{\mathrm{S},1}^{\mathrm{T}} \, \boldsymbol{h}_{\mathrm{S},2}^{\mathrm{T}}\right)^{\mathrm{T}}$; in other words, $\boldsymbol{h}_{\mathrm{E}}$ represents the vector of generalized forces causing the object's motion. From (39.7) and (39.9) it follows that $\boldsymbol{h}_{\mathrm{E}}$ can be expressed in terms of end-effector forces as well

$$\boldsymbol{h}_{\mathrm{E}} = \mathbf{W}_1 \boldsymbol{h}_1 + \mathbf{W}_2 \boldsymbol{h}_2 = \mathbf{W}\boldsymbol{h} \,, \tag{39.10}$$

with $\mathbf{W} = (\mathbf{W}_1 \; \mathbf{W}_2)$ and $\boldsymbol{h} = \left(\boldsymbol{h}_1^{\mathrm{T}} \, \boldsymbol{h}_2^{\mathrm{T}}\right)^{\mathrm{T}}$. It can be recognized that $\mathbf{W}_{\mathrm{S}}$ $(\mathbf{W})$ is a (6×12) matrix, having a six-dimensional (6-D) range space and a six-dimensional

Avidbots Ex1016
Page 1024 of 2154

null space, describing the geometry of the grasp, and thus is usually termed the *grasp matrix*.

The inverse solution to (39.9) yields

$$\boldsymbol{h}_{\mathrm{S}} = \mathbf{W}_{\mathrm{S}}^{\dagger}\,\boldsymbol{h}_{\mathrm{E}} + \mathbf{V}_{\mathrm{S}}\,\boldsymbol{h}_{\mathrm{I}} = \mathbf{U}_{\mathrm{S}}\,\boldsymbol{h}_{\mathrm{O}}\,, \tag{39.11}$$

where $\mathbf{W}_{\mathrm{S}}^{\dagger}$ denotes the Moore–Penrose pseudoinverse of $\mathbf{W}_{\mathrm{S}}$

$$\mathbf{W}_{\mathrm{S}}^{\dagger} = \frac{1}{2}\begin{pmatrix}\mathbf{I}_6\\ \mathbf{I}_6\end{pmatrix}. \tag{39.12}$$

$\mathbf{V}_{\mathrm{S}}$ is a matrix whose columns are a basis of the null space of $\mathbf{W}_{\mathrm{S}}$, e.g.,

$$\mathbf{V}_{\mathrm{S}} = \begin{pmatrix}-\mathbf{I}_6\\ \mathbf{I}_6\end{pmatrix}, \tag{39.13}$$

$\boldsymbol{h}_{\mathrm{O}} = \left(\boldsymbol{h}_{\mathrm{E}}^{\mathrm{T}}\,\boldsymbol{h}_{\mathrm{I}}^{\mathrm{T}}\right)^{\mathrm{T}}$, and

$$\mathbf{U}_{\mathrm{S}} = \left(\mathbf{W}_{\mathrm{S}}^{\dagger}\,\mathbf{V}_{\mathrm{S}}\right). \tag{39.14}$$

The second term on the right-hand side of (39.11), i. e., $\mathbf{V}_{\mathrm{S}}\boldsymbol{h}_{\mathrm{I}}$, represents a vector of generalized forces, referred at the tip of the virtual sticks, which lies in the null space of $\mathbf{W}_{\mathrm{S}}$; thus, such forces do not contribute to external forces. Hence, the (6×1) vector $\boldsymbol{h}_{\mathrm{I}}$ is a generalized force which does not contribute to the object's motion. Therefore, it represents internal loading of the object (i. e., mechanical stresses) and is termed the *internal forces* vector [39.12]. A similar argument can be used for equation (39.10), whose inverse solution is

$$\boldsymbol{h} = \mathbf{W}^{\dagger}\boldsymbol{h}_{\mathrm{E}} + \mathbf{V}\boldsymbol{h}_{\mathrm{I}} = \mathbf{U}\boldsymbol{h}_{\mathrm{O}}\,, \tag{39.15}$$

where

$$\mathbf{U} = (\mathbf{W}^{\dagger}\,\mathbf{V}). \tag{39.16}$$

In [39.13] it has been shown that the first term on the right-hand side of (39.15) represents only contributions to external forces if the pseudoinverse of the grasp matrix is properly defined, i. e.,

$$\mathbf{W}^{\dagger} = \begin{pmatrix}\frac{1}{2}\mathbf{I}_3 & \mathbf{O}_3\\ \frac{1}{2}\mathbf{S}(\boldsymbol{r}_1) & \frac{1}{2}\mathbf{I}_3\\ \frac{1}{2}\mathbf{I}_3 & \mathbf{O}_3\\ \frac{1}{2}\mathbf{S}(\boldsymbol{r}_2) & \frac{1}{2}\mathbf{I}_3\end{pmatrix}. \tag{39.17}$$

As $\mathbf{V}_{\mathrm{S}}$ in (39.11), the columns of $\mathbf{V}$ span the null space of $\mathbf{W}$, and can be chosen as [39.31]

$$\mathbf{V} = \begin{pmatrix}-\mathbf{I}_3 & \mathbf{O}_3\\ -\mathbf{S}(\boldsymbol{r}_1) & -\mathbf{I}_3\\ \mathbf{I}_3 & \mathbf{O}_3\\ \mathbf{S}(\boldsymbol{r}_2) & \mathbf{I}_3\end{pmatrix}. \tag{39.18}$$

A different parametrization of the inverse solutions to (39.9) and (39.10) can be expressed, respectively, as

$$\boldsymbol{h}_{\mathrm{S}} = \mathbf{W}_{\mathrm{S}}^{\dagger}\boldsymbol{h}_{\mathrm{E}} + \left(\mathbf{I}_{12} - \mathbf{W}_{\mathrm{S}}^{\dagger}\mathbf{W}_{\mathrm{S}}\right)\boldsymbol{h}_{\mathrm{S}}^{\star} \tag{39.19}$$

and

$$\boldsymbol{h} = \mathbf{W}^{\dagger}\boldsymbol{h}_{\mathrm{E}} + \left(\mathbf{I}_{12} - \mathbf{W}^{\dagger}\mathbf{W}\right)\boldsymbol{h}^{\star}\,, \tag{39.20}$$

where $\boldsymbol{h}_{\mathrm{S}}^{\star}$ ($\boldsymbol{h}^{\star}$) is an arbitrary (12×1) vector of generalized forces acting at the tip of the i-th virtual stick (i-th end-effector) projected onto the null space of $\mathbf{W}_{\mathrm{S}}$ ($\mathbf{W}$) via $\mathbf{I}_{12} - \mathbf{W}_{\mathrm{S}}^{\dagger}\mathbf{W}_{\mathrm{S}}$ ($\mathbf{I}_{12} - \mathbf{W}^{\dagger}\mathbf{W}$).

By exploiting the principle of virtual work, the mappings between generalized velocities, dual to those derived above, can be established. In detail, the mapping dual to (39.11) is

$$\boldsymbol{v}_{\mathrm{O}} = \mathbf{U}_{\mathrm{S}}^{\mathrm{T}}\,\boldsymbol{v}_{\mathrm{S}}\,, \tag{39.21}$$

where $\boldsymbol{v}_{\mathrm{S}} = \left(\boldsymbol{v}_{\mathrm{S},1}^{\mathrm{T}}\,\boldsymbol{v}_{\mathrm{S},2}^{\mathrm{T}}\right)^{\mathrm{T}}$, $\boldsymbol{v}_{\mathrm{O}} = \left(\boldsymbol{v}_{\mathrm{E}}^{\mathrm{T}}\,\boldsymbol{v}_{\mathrm{I}}^{\mathrm{T}}\right)^{\mathrm{T}}$. The vector $\boldsymbol{v}_{\mathrm{E}}$ can be interpreted as the absolute velocity of the object, while $\boldsymbol{v}_{\mathrm{I}}$ represents the relative velocity between the two coordinate frames $\mathcal{T}_{\mathrm{S},1}$ and $\mathcal{T}_{\mathrm{S},2}$ attached to the tips of the virtual sticks [39.12]; this vector is null when the manipulated object is rigid and rigidly grasped. In a similar way, from (39.15), the following mapping can be devised

$$\boldsymbol{v}_{\mathrm{O}} = \mathbf{U}^{\mathrm{T}}\,\boldsymbol{v}\,, \tag{39.22}$$

where $\boldsymbol{v} = \left(\boldsymbol{v}_1^{\mathrm{T}}\,\boldsymbol{v}_2^{\mathrm{T}}\right)^{\mathrm{T}}$.

Moreover, a set of position and orientation variables corresponding to $\boldsymbol{v}_{\mathrm{E}}$ and $\boldsymbol{v}_{\mathrm{I}}$ can be defined as follows [39.12, 25]

$$\boldsymbol{p}_{\mathrm{E}} = \frac{1}{2}(\boldsymbol{p}_{\mathrm{S},1} + \boldsymbol{p}_{\mathrm{S},2})\,, \quad \boldsymbol{p}_{\mathrm{I}} = \boldsymbol{p}_{\mathrm{S},2} - \boldsymbol{p}_{\mathrm{S},1}\,, \tag{39.23}$$

$$\mathbf{R}_{\mathrm{E}} = \mathbf{R}_1\mathbf{R}^1(\boldsymbol{k}_{21}^1, \vartheta_{21}/2)\,, \quad \mathbf{R}_{\mathrm{I}}^1 = \mathbf{R}_2^1\,, \tag{39.24}$$

where $\mathbf{R}_2^1 = \mathbf{R}_1^{\mathrm{T}}\mathbf{R}_2$ is the matrix expressing the orientation of $\mathcal{T}_2$ with respect to the axes of $\mathcal{T}_1$, while $\boldsymbol{k}_{21}^1$ and ϑ_{21} are, respectively, the equivalent unit vector (expressed with respect to $\mathcal{T}_1$) and rotation angle that determine the mutual orientation expressed by $\mathbf{R}_2^1$. Hence, $\mathbf{R}_{\mathrm{E}}$ expresses a rotation about $\boldsymbol{k}_{21}^1$ by an angle which is half the angle needed to align $\mathcal{T}_2$ with $\mathcal{T}_1$.

In turn, if the orientation variables are expressed in terms of Euler angles, a set of operational space variables can be defined as

$$\boldsymbol{x}_{\mathrm{E}} = \begin{pmatrix}\boldsymbol{p}_{\mathrm{E}}\\ \boldsymbol{\phi}_{\mathrm{E}}\end{pmatrix}, \quad \boldsymbol{x}_{\mathrm{I}} = \begin{pmatrix}\boldsymbol{p}_{\mathrm{I}}\\ \boldsymbol{\phi}_{\mathrm{I}}\end{pmatrix}, \tag{39.25}$$

where

$$\boldsymbol{\phi}_{\mathrm{E}} = \frac{1}{2}(\boldsymbol{\phi}_{\mathrm{S},1} + \boldsymbol{\phi}_{\mathrm{S},2}) \,, \quad \boldsymbol{\phi}_{\mathrm{I}} = \boldsymbol{\phi}_{\mathrm{S},2} - \boldsymbol{\phi}_{\mathrm{S},1} \,. \tag{39.26}$$

It must be remarked, however, that the definitions in (39.26) keep a clear geometric meaning only if the orientation displacements between the virtual stick frames are small. In this case, as shown in [39.11, 12], the corresponding operational space velocities, $\dot{\boldsymbol{x}}_{\mathrm{E}}$ and $\dot{\boldsymbol{x}}_{\mathrm{I}}$, correspond to $\boldsymbol{v}_{\mathrm{E}}$ and $\boldsymbol{v}_{\mathrm{I}}$, respectively, with good approximation. Otherwise, if the orientation displacements become large, the variables defined in (39.26) do not represent any meaningful quantities, and other orientation representations have to be adopted, e.g., the unit quaternion (see Chap. 2 on kinematics for a general introduction to quaternions and Sect. 39.3 for an application to cooperative robots kinematics).

Finally, by using (39.15) and (39.22), together with equations (39.3) and (39.6), the kinetostatic mappings between the force/velocities at the object and their counterparts in the joint space of the manipulators can be given

$$\boldsymbol{\tau} = \mathbf{J}_{\mathrm{O}}^{\mathrm{T}} \boldsymbol{h}_{\mathrm{O}} \,, \tag{39.27}$$

$$\boldsymbol{v}_{\mathrm{O}} = \mathbf{J}_{\mathrm{O}} \dot{\boldsymbol{q}} \,, \tag{39.28}$$

where $\boldsymbol{\tau} = \left(\boldsymbol{\tau}_1^{\mathrm{T}} \; \boldsymbol{\tau}_2^{\mathrm{T}}\right)^{\mathrm{T}}$, $\boldsymbol{q} = \left(\boldsymbol{q}_1^{\mathrm{T}} \; \boldsymbol{q}_2^{\mathrm{T}}\right)^{\mathrm{T}}$, and

$$\mathbf{J}_{\mathrm{O}} = \mathbf{U}^{\mathrm{T}} \mathbf{J} \,, \quad \mathbf{J} = \begin{pmatrix} \mathbf{J}_1 & \mathbf{O}_6 \\ \mathbf{O}_6 & \mathbf{J}_2 \end{pmatrix} \,. \tag{39.29}$$

Formally similar mappings can be established in terms of operational space velocities $\dot{\boldsymbol{x}}_{\mathrm{E}}$ and $\dot{\boldsymbol{x}}_{\mathrm{I}}$ [39.11, 12] and the corresponding operational space forces/moments.

39.2.2 Multifingered Manipulation

Hereafter some connections between the field of multi-arm cooperative manipulation, described in this chapter, and multifingered manipulation, described in Chap. 38, are briefly outlined, focusing on the kinetostatics of the two classes of manipulation systems.

Both in the case of multi-arm cooperative systems and of multifingered manipulation, two or more manipulation structures grasp a commonly manipulated object.

Cooperative manipulation via multiple robotic arms is achieved by rigidly grasping the object (e.g., via rigid fixtures), and thus interaction takes place by exchanging both forces and moments at the grasp points. In other words, all the translational and rotational motion components are transmitted through the grasp points.

On the other side, when a multifingered hand manipulates an object, only some components of the motion are transmitted through the contact points. This is effectively modeled via properly defined constraint matrices, whose expression depends on the type of contact. In other words, constraint matrices act as filters selecting the components of the motion transmitted through each contact. In fact, as shown in Chap. 38, it is convenient to think of the object–finger contact point as twofold: a point on the tip of the finger and a point on the object. Hence, two generalized velocity vectors are defined for the i-th contact point (both expressed with respect to frame $\mathcal{T}_i$): the velocity of the contact point on the hand, $\boldsymbol{v}_{\mathrm{h},i}^{i}$, and the velocity of the contact point on the object $\boldsymbol{v}_{\mathrm{o},i}^{i}$. The corresponding dual generalized force vectors are $\boldsymbol{h}_{\mathrm{h},i}^{i}$ and $\boldsymbol{h}_{\mathrm{o},i}^{i}$, respectively. A contact model is then defined via the $(m_i \times 6)$ constraint matrix $\mathbf{H}_i$, selecting m_i velocity components transmitted through the contact ($\boldsymbol{v}_{\mathrm{t},i}^{i}$), i. e.,

$$\boldsymbol{v}_{\mathrm{t},i}^{i} = \mathbf{H}_i \boldsymbol{v}_{\mathrm{h},i}^{i} = \mathbf{H}_i \boldsymbol{v}_{\mathrm{o},i}^{i} \,. \tag{39.30}$$

The relation dual to (39.30) is

$$\mathbf{H}_i^{\mathrm{T}} \boldsymbol{h}_{\mathrm{t},i}^{i} = \boldsymbol{h}_{\mathrm{h},i}^{i} = \boldsymbol{h}_{\mathrm{o},i}^{i} \,, \tag{39.31}$$

where $\boldsymbol{h}_{\mathrm{t},i}^{i}$ is the vector of the transmitted generalized forces. Hence, (39.10) can be rewritten as

$$\boldsymbol{h}_{\mathrm{E}} = \mathbf{W}_1 \overline{\mathbf{R}}_1 \mathbf{H}_1^{\mathrm{T}} \boldsymbol{h}_{\mathrm{t},1}^{1} + \mathbf{W}_2 \overline{\mathbf{R}}_2 \mathbf{H}_2^{\mathrm{T}} \boldsymbol{h}_{\mathrm{t},2}^{2} \,, \tag{39.32}$$

where $\overline{\mathbf{R}}_i = \mathrm{diag}\{\mathbf{R}_i, \mathbf{R}_i\}$. Hence, a conceptually similar kinetostatic analysis can be developed, leading to the concept of external and internal forces (and related kinematic quantities) as well.

However, it must be remarked that, while in the case of cooperative manipulators, tightly grasping the object, internal stresses are usually undesirable effects (unless controlled squeezing of a deformable object has to be achieved), in multifingered hands suitably controlled internal forces are useful to guarantee a firm grasp even in the presence of external loading applied to the object (see the problem of form and force closure discussed in Chap. 38).

Avidbots Ex1016
Page 1026 of 2154

39.3 Cooperative Task Space

The symmetric formulation reviewed in Sect. 39.2.1 defines the fundamental variables and the primary relationships needed to describe the kinematics and the statics of the cooperative manipulators. However, the symmetric formulation was not originally conceived for task and motion planning of cooperative robotic systems, since this formulation effectively describes a cooperative system in terms of forces and velocities at the virtual sticks, while the user is expected to plan the task in terms of geometrically meaningful variables describing the position and the orientation of a set of relevant coordinate frames. Thus, starting from equations (39.23) and (39.24), further research efforts have been devoted to define alternative kinematic formulations, explictly aimed at planning the coordinated motion of general multi-arm systems. A notable example of such alternative formulations is the task-oriented formulation for the definition of the *cooperative task space* originally proposed in [39.25, 26]. By starting from (39.23) and (39.24), this formulation defines directly the task variables in terms of the *absolute* and *relative* motion of the cooperative system, which can be directly computed from the position/orientation of the end-effector coordinate frames.

Let us define as *absolute coordinate frame*, $\mathcal{T}_a$, the frame whose position in the base frame is given by the position vector $\boldsymbol{p}_a$ (*absolute position*)

$$\boldsymbol{p}_a = \frac{1}{2}(\boldsymbol{p}_1 + \boldsymbol{p}_2)\,. \tag{39.33}$$

The orientation of $\mathcal{T}_a$ with respect to the base frame (*absolute orientation*) is defined by means of the rotation matrix $\mathbf{R}_a$

$$\mathbf{R}_a = \mathbf{R}_1\mathbf{R}^1\left(\boldsymbol{k}_{21}^1, \vartheta_{21}/2\right). \tag{39.34}$$

Of course, the sole absolute variables cannot uniquely specify the cooperative motion, since 12 variables are required, e.g., for a two-arm system. Hence, in order to devise a complete description of the system's configuration, the position/orientation of each manipulator relative to the other manipulators in the system must be considered. In detail, for a two-arm system, the *relative position* between the manipulators is defined as

$$\boldsymbol{p}_r = \boldsymbol{p}_2 - \boldsymbol{p}_1, \tag{39.35}$$

while the *relative orientation* between the two end-effector frames is expressed via the following rotation matrix

$$\mathbf{R}_r^1 = \mathbf{R}_1^{\mathrm{T}}\mathbf{R}_2 = \mathbf{R}_2^1\,. \tag{39.36}$$

The variables $\boldsymbol{p}_a$, $\mathbf{R}_a$, $\boldsymbol{p}_r$, and $\mathbf{R}_r^1$ define the *cooperative task space*. Remarkably, $\mathbf{R}_a$ and $\mathbf{R}_r^1$ coincide with $\mathbf{R}_E$ and $\mathbf{R}_I^1$, respectively.

It is worth pointing out a useful feature of the above defined cooperative task space formulation. In fact, it can be recognized that definitions (39.33)–(39.36) are not based on any special assumption on the nature of the manipulated object and/or the grasp. In other words, the cooperative task space variables can be effectively used to describe cooperative systems manipulating non-rigid objects and/or characterized by a non-rigid grasp. Also, the same task-space variables can be used to describe a pure motion coordination task, i. e., when the arms are required to perform a coordinated motion without physically interacting via a commonly manipulated object. When the manipulators hold a rigid object (or a deformable object for which deformations are not commanded), then relative position and orientation are to be kept constant. Otherwise, if a relative motion between the end-effectors is allowed, then $\boldsymbol{p}_r^1$ and $\mathbf{R}_r^1$ may vary according to the actual relative motion.

As shown in [39.25, 26], absolute

$$\dot{\boldsymbol{p}}_a = \frac{1}{2}(\dot{\boldsymbol{p}}_1 + \dot{\boldsymbol{p}}_2)\,, \quad \boldsymbol{\omega}_a = \frac{1}{2}(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2)\,, \tag{39.37}$$

and relative

$$\dot{\boldsymbol{p}}_r = (\dot{\boldsymbol{p}}_2 - \dot{\boldsymbol{p}}_1)\,, \quad \boldsymbol{\omega}_r = \boldsymbol{\omega}_2 - \boldsymbol{\omega}_1\,, \tag{39.38}$$

linear and angular velocities can be readily derived, as well as their dual variables (i. e., absolute/relative forces and moments)

$$\boldsymbol{f}_a = \boldsymbol{f}_1 + \boldsymbol{f}_2\,, \quad \boldsymbol{n}_a = \boldsymbol{n}_1 + \boldsymbol{n}_2\,, \tag{39.39}$$

$$\boldsymbol{f}_r = \frac{1}{2}(\boldsymbol{f}_2 - \boldsymbol{f}_1)\,, \quad \boldsymbol{n}_r = \frac{1}{2}(\boldsymbol{n}_2 - \boldsymbol{n}_1)\,. \tag{39.40}$$

Kinetostatic mappings, analogous to those derived in the previous section, can be established between linear/angular velocities (force/moments) and their counterparts defined at the end-effectors (or joints) level of each manipulator [39.25, 26].

Remarkably, the variables defined by the symmetric formulation and those used in the task-oriented formulation are related via simple mappings. In fact, forces (angular velocities, orientation variables) always coincide in the two formulations, while moments (linear velocities, position variables) coincide only when the object reduces to a point, or when the kinematics of each manipulator is referred to the tip of the corresponding virtual stick.

Avidbots Ex1016
Page 1027 of 2154

As shown in the following example, in the case of planar cooperative systems, the definition of the cooperative task-space variables is straightforward.

Example 39.1 (Cooperative Task-Space Variables for a Planar Two-Arm System)
For a planar two-arm system the i-th end-effector coordinates can be defined via the (3×1) vector

$$\boldsymbol{x}_i = \begin{pmatrix} \boldsymbol{p}_i \\ \varphi_i \end{pmatrix}, \quad i = 1, 2 \,,$$

where $\boldsymbol{p}_i$ is the (2×1) vector of the i-th end-effector position in the plane and φ_i is the angle describing its orientation (i. e., the rotation of the end-effector frame about an axis orthogonal to the plane). Hence, the task-space variables can be readily defined as

$$\boldsymbol{x}_\mathrm{a} = \frac{1}{2}(\boldsymbol{x}_1 + \boldsymbol{x}_2) \,, \tag{39.41}$$

$$\boldsymbol{x}_\mathrm{r} = \boldsymbol{x}_2 - \boldsymbol{x}_1 \,, \tag{39.42}$$

since the orientation of each end-effector is simply represented by an angle.

In the spatial case, the definition of the orientation variables in terms of Euler angles, as in (39.34) and (39.36), is somewhat critical; in fact, since $\mathcal{T}_1$ and $\mathcal{T}_2$ do not coincide in general, orientation displacements are likely to be large and definitions analogous to (39.26) are not correct. Hence, geometrically meaningful orientation representations have to be adopted, e.g., the unit quaternion (Chap. 2). In detail, by following the approach in [39.26], the orientation variables can be defined as follows. Let

$$\mathcal{Q}^1_{\mathrm{k}>} = \{\eta_\mathrm{k}, \boldsymbol{\epsilon}^1_\mathrm{k}\} = \left\{\cos\frac{\vartheta_{21}}{4}, \boldsymbol{k}^1_{21}\sin\frac{\vartheta_{21}}{4}\right\} \tag{39.43}$$

denote the unit quaternion extracted from $\mathbf{R}^1(\boldsymbol{k}^1_{21}, \vartheta_{21}/2)$; let, also, $\mathcal{Q}_1 = \{\eta_1, \boldsymbol{\epsilon}_1\}$ and $\mathcal{Q}_2 = \{\eta_2, \boldsymbol{\epsilon}_2\}$ denote the unit quaternions extracted from $\mathbf{R}_1$ and $\mathbf{R}_2$, respectively. Then, the absolute orientation can be expressed in terms of quaternion product as follows

$$\mathcal{Q}_\mathrm{a} = \{\eta_\mathrm{a}, \boldsymbol{\epsilon}_\mathrm{a}\} = \mathcal{Q}_1 * \mathcal{Q}^1_\mathrm{k} \,, \tag{39.44}$$

while the relative orientation can be expressed as the quaternion product

$$\mathcal{Q}^1_\mathrm{r} = \{\eta_\mathrm{r}, \boldsymbol{\epsilon}^1_\mathrm{r}\} = \mathcal{Q}_1^{-1} * \mathcal{Q}_2 \,, \tag{39.45}$$

where $\mathcal{Q}_1^{-1} = \{\eta_1, -\boldsymbol{\epsilon}_1\}$ (i. e., the conjugate of $\mathcal{Q}_1$) represents the unit quaternion extracted from $\mathbf{R}_1^\mathrm{T}$.

Finally, it is worth mentioning that a more general cooperative task-space formulation is developed for different classes of multi-arm robotic systems in [39.30].

39.4 Dynamics and Load Distribution

The equations of motion of the i-th manipulator in a cooperative manipulation system are given by

$$\mathbf{M}_i(\boldsymbol{q}_i)\,\ddot{\boldsymbol{q}}_i + \boldsymbol{c}_i(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) = \boldsymbol{\tau}_i - \mathbf{J}_i^\mathrm{T}(\boldsymbol{q}_i)\boldsymbol{h}_i \,, \tag{39.46}$$

where $\mathbf{M}_i(\boldsymbol{q}_i)$ is the symmetric positive-definite inertia matrix and $\boldsymbol{c}_i(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$ is the vector of the forces/torques due to the centrifugal, Coriolis, gravity, and friction effects. The model can be expressed in compact form as

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} - \mathbf{J}^\mathrm{T}(\boldsymbol{q})\boldsymbol{h} \,, \tag{39.47}$$

where the matrices are block-diagonal (e.g., $\mathbf{M} = \mathrm{blockdiag}\{\mathbf{M}_1, \mathbf{M}_2\}$) and the vectors are stacked (e.g., $\boldsymbol{q} = (\boldsymbol{q}_1^\mathrm{T}\ \boldsymbol{q}_2^\mathrm{T})^\mathrm{T}$).

The object's motion is described by the classical Newton–Euler equations of rigid body

$$\mathbf{M}_\mathrm{E}(\mathbf{R}_\mathrm{E})\,\dot{\boldsymbol{v}}_\mathrm{E} + \boldsymbol{c}_\mathrm{E}(\mathbf{R}_\mathrm{E}, \boldsymbol{\omega}_\mathrm{E})\boldsymbol{v}_\mathrm{E} = \boldsymbol{h}_\mathrm{E} = \mathbf{W}\boldsymbol{h} \,, \tag{39.48}$$

where $\mathbf{M}_\mathrm{E}$ is the inertia matrix of the object and $\boldsymbol{c}_\mathrm{E}$ collects the nonlinear components of the inertial forces/moments (i. e., gravity, centrifugal and Coriolis forces/moments).

The above equations must be completed by imposing the closed-chain constraints arising from the kinematic coupling between the two manipulators through the commonly manipulated rigid object. The constraints can be expressed by imposing a null internal velocity vector in the mapping (39.21)

$$\boldsymbol{v}_\mathrm{I} = \mathbf{V}_\mathrm{S}^\mathrm{T}\boldsymbol{v}_\mathrm{S} = \boldsymbol{v}_{\mathrm{S},1} - \boldsymbol{v}_{\mathrm{S},2} = \boldsymbol{0} \,, \tag{39.49}$$

which can be expressed, by using (39.8) and (39.22), in terms of end-effector velocities (where the notation $\mathbf{W}_i^{-\mathrm{T}}$ stands for $(\mathbf{W}_i^\mathrm{T})^{-1}$)

$$\mathbf{V}^\mathrm{T}\boldsymbol{v} = \mathbf{W}_1^{-\mathrm{T}}\boldsymbol{v}_1 - \mathbf{W}_2^{-\mathrm{T}}\boldsymbol{v}_2 = \boldsymbol{0} \,, \tag{39.50}$$

and, finally, in terms of joint velocities

$$\mathbf{V}^\mathrm{T}\mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{W}_1^{-\mathrm{T}}\mathbf{J}_1(\boldsymbol{q}_1)\dot{\boldsymbol{q}}_1 - \mathbf{W}_2^{-\mathrm{T}}\mathbf{J}_2(\boldsymbol{q}_2)\dot{\boldsymbol{q}}_2 = \boldsymbol{0} \,. \tag{39.51}$$

Equations (39.47), (39.48), and (39.51) represent a constrained dynamical model of the cooperative system in the joint space; $N_1 + N_2$ generalized coordinates (i. e., $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$) are related to each other by the six algebraic closed-chain constraints (39.51). This implies that the total number of degrees of freedom is $N_1 + N_2 - 6$ and the model has the form of a set of differential-algebraic equations.

39.4.1 Reduced-Order Models

The above derived dynamic model incorporates a set of closed-chain constraints, which reduces the number of independent generalized coordinates to $N_1 + N_2 - 6$. Hence, it is expected that, by eliminating six equations via the constraints (39.51), a reduced-order model can be obtained. Early work on reduced-order modeling of closed chains can be, e.g., found in [39.67]. Later, in [39.68, 69], this problem has been tackled by first deriving, from (39.47), (39.48), and (39.51), a joint space model of the whole closed chain in the form

$$\mathbf{M}_{\mathrm{C}}(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \boldsymbol{c}_{\mathrm{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{D}_{\mathrm{C}}(\boldsymbol{q})\boldsymbol{\tau}\,, \tag{39.52}$$

where $\mathbf{M}_{\mathrm{C}}$, $\mathbf{D}_{\mathrm{C}}$, and $\boldsymbol{c}_{\mathrm{C}}$ depend on the dynamics of the manipulators and the object, as well as on the geometry of the grasp. The above model can be integrated, once the joint torques vector $\boldsymbol{\tau}$ is specified over an assigned time interval, to solve for the joint variables $\boldsymbol{q}$ (*forward dynamics*). However, the model cannot be used to find $\boldsymbol{\tau}$ from assigned $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\ddot{\boldsymbol{q}}$ (*inverse dynamics*), since the $((N_1 + N_2) \times (N_1 + N_2))$ matrix $\mathbf{D}_{\mathrm{C}}$ is not full rank [39.69], and thus the inverse dynamics problem turns out to be underspecified.

In order to find a reduced-order model, composed by $N_1 + N_2 - 6$ equations, a $((N_1 + N_2 - 6) \times 1)$ pseudovelocity vector has to be considered

$$\boldsymbol{\nu} = \mathbf{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\,, \tag{39.53}$$

where the $((N_1 + N_2 - 6) \times (N_1 + N_2))$ matrix $\mathbf{B}(\boldsymbol{q})$ is selected so that $(\mathbf{A}^{\mathrm{T}}(\boldsymbol{q})\ \mathbf{B}^{\mathrm{T}}(\boldsymbol{q}))^{\mathrm{T}}$ is nonsingular and

$$\mathbf{A}(\boldsymbol{q}) = \mathbf{W}_2^{\mathrm{T}}\,\mathbf{V}^{\mathrm{T}}(\boldsymbol{q})\,.$$

Then, the reduced-order model can be written in terms of the variables $\boldsymbol{q}$, $\boldsymbol{\nu}$, and $\dot{\boldsymbol{\nu}}$ as

$$\boldsymbol{\Sigma}^{\mathrm{T}}(\boldsymbol{q})\mathbf{M}_{\mathrm{C}}(\boldsymbol{q})\boldsymbol{\Sigma}(\boldsymbol{q})\,\dot{\boldsymbol{\nu}} + \boldsymbol{\Sigma}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{c}_{\mathrm{R}}(\boldsymbol{q}, \boldsymbol{\nu}) = \boldsymbol{\Sigma}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\tau}\,, \tag{39.54}$$

where $\boldsymbol{\Sigma}$ is an $((N_1 + N_2) \times (N_1 + N_2 - 6))$ matrix such that

$$\begin{pmatrix}\mathbf{A}\\ \mathbf{B}\end{pmatrix}^{-1} = (\boldsymbol{\Pi}\ \boldsymbol{\Sigma})\,,$$

and $\boldsymbol{c}_{\mathrm{R}}$ depends on $\boldsymbol{c}_{\mathrm{C}}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\Pi}$ [39.69]. The reduced-order model can be used for computing the forward dynamics; in this case, however, the problem of expressing a reduced set of *pseudocoordinates* related to $\boldsymbol{\nu}$ in the numerical integration has to be considered. Since $\boldsymbol{\Sigma}^{\mathrm{T}}$ is nonsquare, the inverse dynamics problem still admits infinitely many solutions in terms of $\boldsymbol{\tau}$. However, this does not prevent the application model (39.54) to cooperative manipulators control (e.g., the decoupled control architecture proposed in [39.68, 69]).

39.4.2 Load Distribution

The problem of load sharing in multi-arm robotic systems is that of distributing the load among the arms composing the system (e.g., a strong arm may share the load more than a weak one). This is possible because a multi-arm system has redundant actuators; in fact, if a robotic arm is equipped with the number of actuators strictly needed for supporting the load, no optimization of load distribution is possible. In this section, this problem is presented according to the results in [39.36–42].

We can introduce a load-sharing matrix in the framework adopted for presenting the kinematics of the cooperative manipulators. By replacing the Moore–Penrose inverse in equation (39.11) with a suitably defined generalized inverse, $\mathbf{W}_{\mathrm{S}}^{-}$, the following expression can be devised for the generalized forces at the tip of the virtual sticks

$$\boldsymbol{h}_{\mathrm{S}} = \mathbf{W}_{\mathrm{S}}^{-}\,\boldsymbol{h}_{\mathrm{E}} + \mathbf{V}_{\mathrm{S}}\,\boldsymbol{h}_{\mathrm{I}}'\,, \tag{39.55}$$

where

$$\mathbf{W}_{\mathrm{S}}^{-} = \begin{pmatrix}\mathbf{L}\\ \mathbf{I}_6 - \mathbf{L}\end{pmatrix}^{\mathrm{T}}\,, \tag{39.56}$$

and the matrix $\mathbf{L}$ is the so-called load sharing matrix. It can be easily proved that the nondiagonal elements of $\mathbf{L}$ only yield a $\boldsymbol{h}_{\mathrm{S}}$ vector in the null space of $\mathbf{W}_{\mathrm{S}}$, that is, the space of internal forces/moments. Therefore, without losing generality, let us choose $\mathbf{L}$ such that

$$\mathbf{L} = \mathrm{diag}\,\{\boldsymbol{\lambda}\}\,, \tag{39.57}$$

where the vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_6)^{\mathrm{T}}$ collects a set of constants λ_i representing the load-sharing coefficients.

The problem is that of properly tuning the load-sharing coefficients to ensure correct cooperative manipulation of the object by the arms. In order to answer this question, it must be noticed that, by combining equations (39.11) and (39.55), the following relation can be obtained

$$\boldsymbol{h}_{\mathrm{I}} = \mathbf{V}_{\mathrm{S}}^{\dagger}\left(\mathbf{W}_{\mathrm{S}}^{-} - \mathbf{W}_{\mathrm{S}}^{\dagger}\right)\boldsymbol{h}_{\mathrm{E}} + \boldsymbol{h}_{\mathrm{I}}'\,, \tag{39.58}$$

Avidbots Ex1016
Page 1029 of 2154

which, bearing in mind that only $\boldsymbol{h}_E$ and $\boldsymbol{h}_S$ are really existing forces/moments, indicates that:

- $\boldsymbol{h}_I$, $\boldsymbol{h}'_I$, and λ_i can be thought of as *artificial* parameters, introduced for better understanding of the manipulation process.
- $\boldsymbol{h}'_I$ and λ_i are not independent; the concept of internal forces/moments and the concept of load sharing are mathematically mixed with each other.

Therefore, tuning the load-sharing coefficients or choosing suitable internal forces/moments is strictly equivalent from the mathematical (and, also, from the performance) point of view. Only one variable among $\boldsymbol{h}_I$, $\boldsymbol{h}'_I$, and $\boldsymbol{\lambda}$ is independent; hence, any of those redundant parameters, to be optimized for load sharing, can be exploited. This is more generally stated in [39.39, 40]. In [39.42], internal forces/moments $\boldsymbol{h}_I$ are tuned, both for the sake of simplicity and for consistency with the adopted control laws.

A very relevant problem related to load sharing is that of robust holding, i. e., the problem of determining forces/moments applied to object by the arms, $\boldsymbol{h}_S$, required to keep the grasp even in the presence of disturbing external forces/moments. This problem can be solved by tuning the internal forces/moments (or the load-sharing coefficients, of course) and is addressed, e.g., in [39.41], where conditions to keep the grasp are expressed via end-effector forces/moments. Namely, by replacing $\boldsymbol{h}_S$ in (39.55) into the equations expressing the grasp conditions, a set of linear inequalities for both $\boldsymbol{h}'_I$ and $\boldsymbol{\lambda}$ are obtained

$$\mathbf{A}_L \boldsymbol{h}'_I + \mathbf{B}_L \boldsymbol{\lambda} < \boldsymbol{c}_L \,, \tag{39.59}$$

where $\mathbf{A}_L$ and $\mathbf{B}_L$ are 6×6 matrices and $\boldsymbol{c}_L$ is a (6×1) vector. In [39.41], a solution $\boldsymbol{\lambda}$ for the inequality is obtained heuristically. The above inequality can be transformed into another inequality expressed with respect to $\boldsymbol{h}_I$, of course; however, $\boldsymbol{\lambda}$ is fitter to such heuristic algorithm because it can be understood intuitively. The same problem may be solved mathematically, by introducing an objective function to be optimized; in this way, the problem is recast in the framework of mathematical programming. To this purpose, let us choose a quadratic cost function of $\boldsymbol{h}_I$ to be minimized

$$f = \boldsymbol{h}_I^T \mathbf{Q} \boldsymbol{h}_I \,, \tag{39.60}$$

where $\mathbf{Q}$ is a (6×6) positive-definite matrix. The above cost function can be seen as a (pseudo)energy to be dissipated by the joint actuators: i. e., the arms dissipate electrical energy in the actuators to yield the internal forces/moments $\boldsymbol{h}_I$. A solution to the quadratic programming problem (39.60) can be found, e.g., in [39.42].

For further insights on the problem of robust holding in the framework of multifingered manipulation, the reader is referred to Chap. 38.

Interestingly, in [39.37] the problem of load distribution is formulated in such a way to take into account manipulators dynamics at the joint level. This approach allows the expression of the load-sharing problem directly in terms of joints actuators torques; at this level, different subtask performance indexes can be used, in a similar way as was done to solve the inverse kinematics of redundant manipulators, to achieve load distribution solutions.

39.5 Task-Space Analysis

As for single-arm robotic systems, a major issue in cooperative manipulation is that of task-space performance evaluation via the definition of suitable manipulability ellipsoids. These concepts have been extended to multi-arm robotic systems in [39.31]. Namely, by exploiting the kinetostatic formulation in Sect. 39.3, velocity and force manipulability ellipsoids are defined by regarding the whole cooperative system as a mechanical transformer from the joint space to the cooperative task space. Since the construction of force/velocity ellipsoids involves nonhomogeneous quantities (i. e., forces and moments, linear and angular velocities), special attention must be paid to the definition of such concepts [39.14, 70, 71]. Also, as for the ellipsoids involving internal forces, a major issue is that of physically meaningful parametrization of the internal forces (see, e.g., the work in [39.15, 16]).

In detail, by following the approach in [39.31], the *external force manipulability ellipsoid* can be defined by the following scalar equation

$$\boldsymbol{h}_E^T \left(\mathbf{J}_E \mathbf{J}_E^T \right) \boldsymbol{h}_E = 1 \,, \tag{39.61}$$

where $\mathbf{J}_E = \mathbf{W}^{\dagger T} \mathbf{J}$. The *external velocity manipulability ellipsoid* is defined dually by the following scalar equation

$$\boldsymbol{v}_E^T \left(\mathbf{J}_E \mathbf{J}_E^T \right)^{-1} \boldsymbol{v}_E = 1 \,. \tag{39.62}$$

In the case of dual-arm systems the *internal force manipulability ellipsoid* can be defined as

$$\boldsymbol{h}_I^T \left(\mathbf{J}_I \mathbf{J}_I^T \right) \boldsymbol{h}_I = 1 \,, \tag{39.63}$$

Avidbots Ex1016
Page 1030 of 2154

where $\mathbf{J}_\mathrm{I} = \mathbf{V}^\mathrm{T}\mathbf{J}$. Also, the *internal velocity ellipsoid* can be defined, via kinetostatic duality, as

$$\boldsymbol{v}_\mathrm{I}^\mathrm{T}\left(\mathbf{J}_\mathrm{I}\mathbf{J}_\mathrm{I}^\mathrm{T}\right)^{-1}\boldsymbol{v}_\mathrm{I} = 1\,. \tag{39.64}$$

The internal force/velocity ellipsoids can be defined in the case of cooperative systems composed of more than two manipulators by considering one pair of interacting end-effectors at a time [39.31].

The manipulability ellipsoids can be seen as performance measures aimed at determining the arms' attitude to cooperate in a given system's configuration. Also, as for the single-arm systems, the manipulability ellipsoids can be used to determine optimal postures for redundant multi-arm systems.

Besides the above described approach, two other main approaches have been proposed to analyze the manipulability of cooperative multi-arm systems: the task-oriented manipulability measure [39.32] and polytopes [39.33]. Moreover, in [39.34] a systematic approach to perform dynamic analysis of multi-arm systems is presented. Namely, the concept of dynamic manipulability ellipsoid is extended to multi-arm systems, and indexes of the system's capability of generating object accelerations along assigned task-space directions are devised.

39.6 Control

When a cooperative multi-arm system is employed for the manipulation of a common object, it is important to control both the absolute motion of the held object and the internal stresses applied to it. Hence, most of the control approaches to cooperative robotic systems can be classified as force/motion control schemes, in that they decompose the control action in a motion control loop, aimed at tracking of the desired object motion, and a force control loop, aimed at controlling the internal loading of the object.

Early approaches to the control of cooperative systems are based on the *master/slave* concept [39.2]. Namely, the cooperative system is decomposed into:

- A master arm, which is in charge of imposing the absolute motion of the object; hence, the master arm is position controlled so as to achieve accurate and robust tracking of position/orientation reference trajectories, in the face of external disturbances (e.g., forces due to the interaction with the other cooperating arms); in other words, the master arm is controlled so as to have a *stiff* behavior.
- The slave arms, which are force controlled so as to achieve a *compliant* behavior with respect to the interaction forces; hence, it is expected that the slave arms are capable of following (as smoothly as possible) the motion imposed by the master arm.

A natural evolution of the above approach is the so-called *leader–follower* [39.67], where the follower arms reference motion is computed via closed-chain constraints.

However, such approaches suffered from implementation issues, mainly due to the fact that the compliance of the slave arms has to be very large, so as to follow the motion imposed by the master arm smoothly. Also, a difficulty arises when the roles of master and slave have to be assigned to the arms for a given cooperative task, since the master/slave modes may need to be dynamically changed during the task execution.

Hence, a more natural non-master/slave approach has been pursued later, where the cooperative system is seen as a whole. Namely, the reference motion of the object is used to determine the motion of all the arms in the system and the interaction forces acting at each end-effector are fed back so as to be directly controlled. To this aim, the mappings between the forces and velocities at the end-effector of each manipulator and their counterparts at the manipulated object are to be considered in the design of the control laws.

39.6.1 Hybrid Control

In [39.11, 12] a non-master/slave approach has been proposed, based on the well-known scheme proposed by Raibert and Craig for the robot/environment interaction control of single-arm systems (Chap. 9). The operational space vector for the hybrid position/force control is defined via the following variables

$$\boldsymbol{x}_\mathrm{O} = \begin{pmatrix}\boldsymbol{x}_\mathrm{E}\\ \boldsymbol{x}_\mathrm{I}\end{pmatrix}\,, \tag{39.65}$$

where $\boldsymbol{x}_\mathrm{E}$, $\boldsymbol{x}_\mathrm{I}$ are the operational space vectors, defined by specifying the orientation via a minimal set of orientation angles (e.g., Euler angles). The generalized forces vector to be considered is

$$\boldsymbol{h}_\mathrm{O} = \begin{pmatrix}\boldsymbol{h}_\mathrm{E}\\ \boldsymbol{h}_\mathrm{I}\end{pmatrix}\,. \tag{39.66}$$

The organization of the control scheme is shown diagrammatically in Fig. 39.2. The suffixes d and m

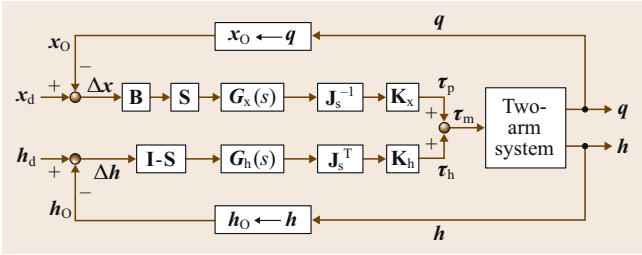


Fig. 39.2 A hybrid position/force control scheme

represent the desired value and the control command, respectively. The command vector $\boldsymbol{\tau}_m$ to the actuators of the two arms is given by two contributions

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_p + \boldsymbol{\tau}_h \,. \tag{39.67}$$

The first term, $\boldsymbol{\tau}_p$, is the command vector for the position control and is given by

$$\boldsymbol{\tau}_p = \mathbf{K}_x \mathbf{J}_s^{-1} \boldsymbol{G}_x(s) \mathbf{S} \mathbf{B} (\boldsymbol{x}_{O,d} - \boldsymbol{x}_O) \,, \tag{39.68}$$

while $\boldsymbol{\tau}_h$ is the command vector for the force control

$$\boldsymbol{\tau}_h = \mathbf{K}_h \mathbf{J}_s^T \boldsymbol{G}_h(s) (\mathbf{I} - \mathbf{S}) (\boldsymbol{h}_{O,d} - \boldsymbol{h}_O) \,. \tag{39.69}$$

The matrix $\mathbf{B}$ transforms the errors on the orientation angles into equivalent rotation vectors. The matrix $\mathbf{J}_s$ is the Jacobian matrix that transforms the joint velocity $\dot{\boldsymbol{q}}$ into the task-space velocity $\boldsymbol{v}_O$. The matrix operators $\boldsymbol{G}_x(s)$ and $\boldsymbol{G}_h(s)$ represent position and force control laws, respectively. The gain matrices $\mathbf{K}_x$ and $\mathbf{K}_h$ are assumed to be diagonal; their diagonal elements convert velocity and force commands into actuator commands, respectively. The matrix $\mathbf{S}$ selects the position-controlled variables; it is diagonal and its diagonal entries take the values of 1 or 0; namely, the i-th workspace coordinate is position controlled if the i-th diagonal element of $\mathbf{S}$ is 1, while it is force controlled if it is 0. Finally, $\mathbf{I}$ is the identity matrix having the same dimensions as $\mathbf{S}$, while $\boldsymbol{q}$ and $\boldsymbol{h}$ are the vectors of measured joint variables and measured end-effector generalized forces, respectively.

39.6.2 PD Force/Motion Control

In [39.17] a Lyapunov-based approach is pursued to devise force/position PD-type control laws. Namely, the joints torques inputs to each arm are computed as the combination of two contributions

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_p + \boldsymbol{\tau}_h \,, \tag{39.70}$$

where $\boldsymbol{\tau}_p$ is a PD-type term (eventually including a feedback/feedforward model-based compensation term), taking care of position control, while $\boldsymbol{\tau}_h$ is in charge of internal forces/moments control.

Namely, the PD and model-based terms can be computed at the joint level

$$\boldsymbol{\tau}_p = \mathbf{K}_p \boldsymbol{e}_q - \mathbf{K}_d \dot{\boldsymbol{q}} + \boldsymbol{g} + \mathbf{J}^T \mathbf{W}^\dagger \boldsymbol{g}_E \,, \tag{39.71}$$

where $\boldsymbol{e}_q = \boldsymbol{q}_d - \boldsymbol{q}$, $\boldsymbol{q}_d$ is the vector of desired joint variables, $\mathbf{K}_p$ and $\mathbf{K}_d$ are positive-definite matrix gains, $\boldsymbol{g}$ is the vector of the gravitational forces/torques acting at the manipulators joints, $\boldsymbol{g}_E$ is the vector of gravity forces/moments at the manipulated object.

Since the cooperative task is usually assigned in terms of absolute and relative motion, the equivalent end-effector desired trajectories are to be computed by using the closed-chain constraints, as in the following example.

Example 39.2 (Computation of Desired Trajectories for a Planar Two-Arm System)

For the planar two-arm system, the desired trajectories defining the cooperative task are assigned by specifying the absolute, $\boldsymbol{x}_{a,d}(t)$, and relative, $\boldsymbol{x}_{r,d}(t)$, desired motion. Then, the corresponding end-effectors desired trajectories can be computed as

$$\boldsymbol{x}_{1,d}(t) = \boldsymbol{x}_{a,d}(t) - \frac{1}{2}\boldsymbol{x}_{r,d}(t) \,, \tag{39.72}$$

$$\boldsymbol{x}_{2,d}(t) = \boldsymbol{x}_{a,d}(t) + \frac{1}{2}\boldsymbol{x}_{r,d}(t) \,, \tag{39.73}$$

where (39.41) and (39.42) have been exploited.

In the spatial case (39.72) and (39.73) do not hold, since the absolute and relative orientation must be expressed in terms of geometrically meaningful quantities, e.g., via (39.44) and (39.45). However, the same approach may be pursued at the expense of a slightly more complex expressions of the above formulas [39.26].

Once, the desired position/orientation for each end-effector has been obtained, the inverse kinematics of each arm has to be computed to provide the desired joint trajectories, $\boldsymbol{q}_d$, to the control loop; to this aim, e.g., numerical inverse kinematics algorithms may be employed (Chaps. 2 and 10).

Also, a PD-type control law can be expressed in terms of end-effector variables, i. e.,

$$\boldsymbol{\tau}_p = \mathbf{J}^T (\mathbf{K}_p \boldsymbol{e} - \mathbf{K}_v \boldsymbol{v}) - \mathbf{K}_d \dot{\boldsymbol{q}} + \boldsymbol{g} + \mathbf{J}^T \mathbf{W}^\dagger \boldsymbol{g}_E \,, \tag{39.74}$$

where $\boldsymbol{e}$ is the tracking error computed in terms of end-effector position/orientation variables, $\boldsymbol{v}$ is the vector

collecting the end-effector velocities, and $\mathbf{K}_p$, $\mathbf{K}_v$, and $\mathbf{K}_d$ are positive-definite matrix gains.

Finally, the same concept may be exploited to design a PD control law directly in the object space

$$\boldsymbol{\tau}_p = \mathbf{J}^T\mathbf{W}^\dagger(\mathbf{K}_p\boldsymbol{e}_E - \mathbf{K}_v\boldsymbol{v}_E) - \mathbf{K}_d\dot{\boldsymbol{q}} + \boldsymbol{g} + \mathbf{J}^T\mathbf{W}^\dagger\boldsymbol{g}_E \,, \tag{39.75}$$

where $\boldsymbol{e}_E$ is the tracking error computed in terms of object absolute position/orientation variables, $\boldsymbol{v}_E$ is the object's generalized velocity vector, and $\mathbf{K}_p$, $\mathbf{K}_v$, and $\mathbf{K}_d$ are positive-definite matrix gains.

The internal force control term can instead be designed as follows

$$\boldsymbol{\tau}_h = \mathbf{J}^T\mathbf{V}\boldsymbol{h}_{I,c} \,, \tag{39.76}$$

where

$$\boldsymbol{h}_{I,c} = \boldsymbol{h}_{I,d} + \boldsymbol{G}_h(s)(\boldsymbol{h}_{I,d} - \boldsymbol{h}_I) \,, \tag{39.77}$$

$\boldsymbol{G}_h(s)$ is a matrix operator representing a strictly proper linear filter, such that $\mathbf{I} - \boldsymbol{G}_h(s)$ has zeros only in the left half plane, and $\boldsymbol{h}_{I,d}$ is the vector of desired internal forces; the internal force vector can be computed from the vector of measured end-effector forces as $\boldsymbol{h}_I = \mathbf{V}^\dagger\boldsymbol{h}$. A particularly simple choice, ensuring a null error at steady state, is given by

$$\boldsymbol{G}_h(s) = \frac{1}{s}\mathbf{K}_h \,,$$

where $\mathbf{K}_h$ is a positive-definite matrix. Remarkably, when an infinitely rigid object/grasp is considered, preprocessing of the force error via a strictly proper filter is needed to ensure closed-loop stability [39.17]; e.g., if a simple proportional feedback is adopted (i. e., $\boldsymbol{G}_h(s) = \mathbf{K}_h$), the closed loop will be unstable in the presence of an arbitrary small time delay, unless a small (namely, smaller than 1) force gain is adopted. In practice, the closed kinematic chain will be characterized by some elasticity (e.g., due to grippers, end-effector force/torques sensors, joints); in this case, the product between the control gain $\mathbf{K}_h$ and the stiffness of the flexible components has to be chosen sufficiently small to ensure stability.

An interesting extension of the above approach has been given in [39.26, 28], where kinetostatic filtering of the control action is performed so as to filter all the components of the control input which contribute to internal stresses at the object. Namely, the control law (39.71) can be modified by weighting the proportional term ($\mathbf{K}_p\,\boldsymbol{e}_q$) via the filtering matrix

$$\boldsymbol{\phi} = \mathbf{J}^T(\mathbf{W}^\dagger\mathbf{W} + \mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^\dagger)\mathbf{J}^{-T} \,,$$

where the (6×6) diagonal matrix $\boldsymbol{\Sigma} = \mathrm{diag}\{\sigma_i\}$ weights the components of $\mathbf{J}^{-T}\mathbf{K}_p\boldsymbol{e}_q$ in each direction of the subspace of the internal forces via the constant values $0 \le \sigma_i \le 1$. In detail, if $\boldsymbol{\Sigma} = \mathbf{O}_6$ then all these components are completely canceled from the control action, while the choice $\boldsymbol{\Sigma} = \mathbf{I}_6$ leads to the control law (39.71) without kinetostatic filtering. In a similar way, the control laws (39.74) and (39.75) can be modified so as to introduce proper kinetostatic filtering of $\mathbf{K}_p\,\boldsymbol{e}$ and $\mathbf{K}_p\,\boldsymbol{e}_E$, respectively.

39.6.3 Feedback Linearization Approaches

A further improvement of the PD-plus-gravity compensation control approach has been achieved by introducing a full model compensation, so as to achieve feedback/feedforward linearization of the closed-loop system. The feedback linearization approach formulated at the operational space level is the base for the so-called *augmented object* approach [39.72, 73]. In this approach the system is modeled in the operational space as a whole, by suitably expressing its inertial properties via a single augmented inertia matrix $\mathbf{M}_O$. Hence, the dynamics of the cooperative system in the operational space can be written as

$$\mathbf{M}_O(\boldsymbol{x}_E)\ddot{\boldsymbol{x}}_E + \boldsymbol{c}_O(\boldsymbol{x}_E, \dot{\boldsymbol{x}}_E) = \boldsymbol{h}_E \,. \tag{39.78}$$

In (39.78), $\mathbf{M}_O$ and $\boldsymbol{c}_O$ are the operational space terms modeling, respectively, the inertial properties of the whole system (manipulators and object) and the Coriolis, centrifugal, friction, and gravity terms.

In the framework of feedback linearization (formulated in the operational space), the problem of controlling the internal forces can be solved, e.g., by resorting to the *virtual linkage* model [39.16] or according to the scheme proposed in [39.29], i. e.,

$$\begin{aligned}\boldsymbol{\tau} = {} & \mathbf{J}^T\mathbf{W}^\dagger\left[\mathbf{M}_O\left(\ddot{\boldsymbol{x}}_{E,d} + \mathbf{K}_v\dot{\boldsymbol{e}}_E + \mathbf{K}_p\boldsymbol{e}_E\right) + \boldsymbol{c}_O\right] \\ & + \mathbf{J}^T\mathbf{V}\left[\boldsymbol{h}_{I,d} + \mathbf{K}_h\int(\boldsymbol{h}_{I,d} - \boldsymbol{h}_I)\right].\end{aligned} \tag{39.79}$$

The above control law yields a linear and decoupled closed-loop dynamics

$$\begin{aligned}\ddot{\boldsymbol{e}}_E + \mathbf{K}_v\dot{\boldsymbol{e}}_E + \mathbf{K}_p\boldsymbol{e}_E = \mathbf{0} \,, \\ \tilde{\boldsymbol{h}}_I + \mathbf{K}_h\int\tilde{\boldsymbol{h}}_I\,dt = \mathbf{0} \,,\end{aligned} \tag{39.80}$$

where $\tilde{\boldsymbol{h}}_I = \boldsymbol{h}_{I,d} - \boldsymbol{h}_I$. Hence, the closed-loop dynamics guarantees asymptotically vanishing motion and force errors.

39.6.4 Impedance Control

An alternative control approach can be pursued based on the well-known impedance concept (Chap. 9). In fact, when a manipulation system interacts with an external environment and/or other manipulators, large values of the contact forces and moments can be avoided by enforcing a compliant behavior, with suitable dynamic features, of the robotic system. Impedance control schemes have been proposed in the case of cooperative manipulation to control object/environment interaction forces [39.21] or internal forces [39.22]. More recently, an impedance scheme for the control of both external forces and internal forces has been proposed [39.74] (in VIDEO 67 some experimental results of impedance control ar a dual-arm cooperative system are documented).

In detail, the impedance scheme in [39.21] enforces the following mechanical impedance behavior between the object displacements and the forces acting on object due to interaction with the environment

$$\mathbf{M}_{\mathrm{E}}\tilde{\boldsymbol{a}}_{\mathrm{E}} + \mathbf{D}_{\mathrm{E}}\tilde{\boldsymbol{v}}_{\mathrm{E}} + \mathbf{K}_{\mathrm{e}}\boldsymbol{e}_{\mathrm{E}} = \boldsymbol{h}_{\mathrm{env}} \,, \tag{39.81}$$

where:

- $\boldsymbol{e}_{\mathrm{E}}$ represents the vector of the object's displacements between the desired and actual pose
- $\tilde{\boldsymbol{v}}_{\mathrm{E}}$ is the difference between the object's desired and actual generalized velocities
- $\tilde{\boldsymbol{a}}_{\mathrm{E}}$ is the difference between the object's desired and actual generalized accelerations,

and $\boldsymbol{h}_{\mathrm{env}}$ is the generalized force acting on object due to the interaction with the environment. The impedance dynamics is characterized in terms of given positive-definite mass ($\mathbf{M}_{\mathrm{E}}$), damping ($\mathbf{D}_{\mathrm{E}}$), and stiffness ($\mathbf{K}_{\mathrm{E}}$) matrices to be properly chosen so as to achieve the desired compliant behavior of the object.

As for the pose displacements used in (39.81), special attention has to be paid to the orientation variables.

Example 39.3 (External Impedance for a Planar Two-Arm System)

For the planar two-arm system the quantities in (39.81) can be defined in a straightforward way. Namely,

$$\boldsymbol{e}_{\mathrm{E}} = \boldsymbol{x}_{\mathrm{E,d}} - \boldsymbol{x}_{\mathrm{E}} \,, \tag{39.82}$$

$$\tilde{\boldsymbol{v}}_{\mathrm{E}} = \dot{\boldsymbol{x}}_{\mathrm{E,d}} - \dot{\boldsymbol{x}}_{\mathrm{E}} \,, \tag{39.83}$$

$$\tilde{\boldsymbol{a}}_{\mathrm{E}} = \ddot{\boldsymbol{x}}_{\mathrm{E,d}} - \ddot{\boldsymbol{x}}_{\mathrm{E}} \,, \tag{39.84}$$

where $\boldsymbol{x}_{\mathrm{E}}$ is the (3×1) vector collecting the object's position and the orientation, while $\boldsymbol{x}_{\mathrm{E,d}}$ represents its desired counterpart. Hence, $\mathbf{M}_{\mathrm{E}}$, $\mathbf{D}_{\mathrm{E}}$, and $\mathbf{K}_{\mathrm{E}}$ are (3×3) matrices.

In the spatial case, orientation displacements cannot be defined as in (39.82), and geometrically meaningful orientation representations (i. e., rotation matrices and/or angle/axis representations) or in terms of operational space variables (i. e., differences between operational space vectors) have to be adopted [39.74].

The impedance scheme in [39.22], enforces a mechanical impedance behavior between the i-th end-effector displacements and the internal forces, i. e.,

$$\mathbf{M}_{\mathrm{I},i}\,\tilde{\boldsymbol{a}}_i + \mathbf{D}_{\mathrm{I},i}\,\tilde{\boldsymbol{v}}_i + \mathbf{K}_{\mathrm{I},i}\,\boldsymbol{e}_i = \boldsymbol{h}_{\mathrm{I},i} \,, \tag{39.85}$$

where:

- $\boldsymbol{e}_i$ is the vector expressing the displacement of the i-th end-effector between the desired and actual pose
- $\tilde{\boldsymbol{v}}_i$ is the vector expressing the difference between the desired and actual velocities of the i-th end-effector
- $\tilde{\boldsymbol{a}}_i$ is the vector expressing the difference between the desired and actual accelerations of the i-th end-effector,

and $\boldsymbol{h}_{\mathrm{I},i}$ is the contribution of the i-th end-effector to the internal force, i. e., the i-th component of the vector $\mathbf{V}\mathbf{V}^{\dagger}\boldsymbol{h}$. Again, the impedance dynamics is characterized in terms of given positive-definite mass ($\mathbf{M}_{\mathrm{I},i}$), damping ($\mathbf{D}_{\mathrm{I},i}$), and stiffness ($\mathbf{K}_{\mathrm{I},i}$) matrices, to be properly chosen so as to achieve a suitable compliant behavior of the end-effectors with respect to internal forces.

Example 39.4 (Internal Impedance for a Planar Two-Arm System)

For the planar two-arm system the quantities in (39.85) can be defined in a straightforward way as well

$$\boldsymbol{e}_i = \boldsymbol{x}_{i,d} - \boldsymbol{x}_i \,, \tag{39.86}$$

$$\tilde{\boldsymbol{v}}_i = \dot{\boldsymbol{x}}_{i,d} - \dot{\boldsymbol{x}}_i \,, \tag{39.87}$$

$$\tilde{\boldsymbol{a}}_i = \ddot{\boldsymbol{x}}_{i,d} - \ddot{\boldsymbol{x}}_i \,, \tag{39.88}$$

where $\boldsymbol{x}_i$ is the (3×1) vector collecting the position in the plane and the orientation angle of the i-th end-effector, while $\boldsymbol{x}_{i,d}$ represents its desired counterpart. Again, $\mathbf{M}_{\mathrm{I},i}$, $\mathbf{D}_{\mathrm{I},i}$, and $\mathbf{K}_{\mathrm{I},i}$ are (3×3) matrices.

As for the spatial case, orientation displacements in (39.85) cannot be defined as in (39.86), and geometrically meaningful orientation representations (i. e., rotation matrices and/or angle/axis representations) or in terms of operational space variables (i. e., differences between operational space vectors) have to be adopted [39.74].

The above two approaches have been combined in [39.74], where two control loops are designed to enforce an impedance behavior both at the object level (external forces) and at the end-effector level (internal forces).

Avidbots Ex1016
Page 1034 of 2154

39.7 Conclusions and Further Reading

In this chapter, the fundamentals of cooperative manipulation have been presented. A historical overview of the research on cooperative manipulation has been provided. The kinematics and dynamics of robotic arms cooperatively manipulating a tightly grasped rigid object are considered. Special topics, such as the definition of a cooperative task space and the problem of load distribution, have also been touched on. Then, the main control approaches for cooperative systems have been discussed. A few advanced topics related to control of cooperative robots, i. e., advanced nonlinear control as well as the modeling and control of cooperative systems including flexible elements, will be briefly outlined in the following.

Some of the first attempts to cope with uncertainties and disturbances in cooperative robots control focused on adaptive strategies [39.23, 24], in which the unknown parameters are estimated online, based on a suitable linear-in-the-parameters model of the uncertainties. In detail, the approach in [39.23] is aimed at controlling the object motion, the interaction force due to the contact object/environment, and the internal forces; the adaptive control law estimates the unknown model parameters, both of the manipulators and of the object, on the basis of suitable error equations. In [39.24], the adaptive control concept is used to design a decentralized scheme, i. e., a centralized coordinator is not used, for redundant cooperative manipulators.

A recently developed control framework for cooperative systems is the so-called synchronization control [39.51, 52]; in this approach the control problem is formulated in terms of suitably defined errors accounting for motion synchronization between the manipulators involved in the cooperative task. Namely, the key idea in [39.51] is to ensure tracking of the desired trajectory assigned to each manipulator, while synchronizing its motion with other manipulators motion. In [39.52], the problem of synchronizing the motion of multiple manipulators systems is solved by using only position measurements; the synchronization controller consists of a feedback control law and a set of nonlinear observers, and synchronization is ensured by suitably defining the coupling errors between the manipulators' motions.

Recently, research efforts have been spent on intelligent control [39.53–55]. Namely, in [39.53], a semi-decentralized adaptive fuzzy control scheme, with $\mathcal{H}_\infty$ performance in motion and internal force tracking, is proposed; the controller of each robot consists of two parts: a model-based adaptive controller and an adaptive fuzzy logic controller; the model-based adaptive controller handles the nominal dynamics, including a purely parametric uncertainties model, while the fuzzy logic controller is aimed at counteracting the effect of unstructured uncertainties and external disturbances. In [39.54], a decentralized adaptive fuzzy control scheme is proposed, where the control law makes use of a multi-input multi-output fuzzy logic engine and a systematic online adaptation mechanism. The approach has been further extended in [39.55].

Finally it is worth mentioning the efforts invested in the investigation of control strategies using partial state feedback, i. e., only joints position and end-effector forces are fed back to the controller. A recent contribution was provided by the work in [39.56], in which a decentralized control algorithm that achieves asymptotic tracking of desired positions and forces by using a nonlinear observer for velocities was proposed.

Flexibility in a cooperative system may arise, e.g., due to the use of nonrigid grippers to grasp the manipulated object. In fact, the adoption of compliant grippers allows large internal forces to be avoided and, at the same time, achieves safe manipulation of the object, even in the presence of failures and unpredicted contacts with the external environment. In detail, the work in [39.61] develops a non-model-based decentralized control scheme. Namely, a proportional–derivative (PD) position feedback control scheme with gravity compensation, is designed; the PD scheme is capable of regulating the position/orientation of the manipulated object and, simultaneously, achieves damping of the vibrations induced by the compliant grippers; also, a hybrid scheme is adopted to control internal forces along the directions in which the compliance of grippers is too low to ensure limited internal stresses at the manipulated object.

Other research efforts have been focused on handling multibodied objects, or even flexible objects [39.43–45]. Those objects are difficult to handle and, therefore, assembly of those objects in manufacturing industry is not automated. Also, cooperative control of multi-flexible-arm robots has been investigated [39.46–48]. Once the modeling and control problem is solved (Chap. 11), the flexible-arm robot is a robot with many merits [39.49]: it is lightweight, compliant, and hence safe, etc. Combining control methods for flexible-arm robots, such as vibration suppression, with the cooperative control methods presented in this chapter is straightforward [39.46]. Automated object-retrieval operation with a two-flexible-arm robot has been demonstrated in [39.47, 48] and in VIDEO 68.

Finally, cooperative transportation and manipulation of objects via multiple mobile manipulators

Avidbots Ex1016
Page 1035 of 2154

can be considered still an open research subject. In fact, although notable research results have been already devised [39.65, 75–77], the foreseen use of robotic teams in industrial settings (hyperflexible robotic workcells) and/or in collaboration with humans (robotic co-worker concept) raises new challenges related to autonomy and safety of such systems. A recently emerged application scenario is the cooperative transportation of objects via multiple aerial robots [39.66] (see VIDEO 66 for an example of cooperative transportation via multiple unmanned aerial vehicles).

Video-References

VIDEO 66 Cooperative grasping and transportation of objects using multiple UAVs
available from http://handbookofrobotics.org/view-chapter/39/videodetails/66

VIDEO 67 Impedance control for cooperative manipulators
available from http://handbookofrobotics.org/view-chapter/39/videodetails/67

VIDEO 68 Cooperative capturing via flexible manipulators
available from http://handbookofrobotics.org/view-chapter/39/videodetails/68

VIDEO 69 Cooperative grasping and transportation of an object using two industrial manipulators
available from http://handbookofrobotics.org/view-chapter/39/videodetails/69

VIDEO 70 Control of cooperative manipulators in the operational space
available from http://handbookofrobotics.org/view-chapter/39/videodetails/70

References

39.1 S. Fujii, S. Kurono: Coordinated computer control of a pair of manipulators, Proc. 4th IFToMM World Congr. (1975) pp. 411–417

39.2 E. Nakano, S. Ozaki, T. Ishida, I. Kato: Cooperational control of the anthropomorphous manipulator *MELARM*, Proc. 4th Int. Symp. Ind. Robots, Tokyo (1974) pp. 251–260

39.3 K. Takase, H. Inoue, K. Sato, S. Hagiwara: The design of an articulated manipulator with torque control ability, Proc. 4th Int. Symp. Ind. Robots, Tokyo (1974) pp. 261–270

39.4 S. Kurono: Cooperative control of two artificial hands by a mini-computer, Prepr. 15th Jt. Conf. Autom. Control (1972) pp. 365–366, (in Japanese)

39.5 A.J. Koivo, G.A. Bekey: Report of workshop on coordinated multiple robot manipulators: planning, control, and applications, IEEE J. Robotics Autom. **4**(1), 91–93 (1988)

39.6 P. Dauchez, R. Zapata: Co-ordinated control of two cooperative manipulators: The use of a kinematic model, Proc. 15th Int. Symp. Ind. Robots, Tokyo (1985) pp. 641–648

39.7 N.H. McClamroch: Singular systems of differential equations as dynamic models for constrained robot systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 21–28

39.8 T.J. Tarn, A.K. Bejczy, X. Yun: New nonlinear control algorithms for multiple robot arms, IEEE Trans. Aerosp. Electron. Syst. **24**(5), 571–583 (1988)

39.9 S. Hayati: Hybrid position/force control of multi-arm cooperating robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 82–89

39.10 M. Uchiyama, N. Iwasawa, K. Hakomori: Hybrid position/force control for coordination of a two-arm robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1242–1247

39.11 M. Uchiyama, P. Dauchez: A symmetric hybrid position/force control scheme for the coordination of two robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Philadelphia (1988) pp. 350–356

39.12 M. Uchiyama, P. Dauchez: Symmetric kinematic formulation and non-master/slave coordinated control of two-arm robots, Adv. Robotics **7**(4), 361–383 (1993)

39.13 I.D. Walker, R.A. Freeman, S.I. Marcus: Analysis of motion and internal force loading of objects grasped by multiple cooperating manipulators, Int. J. Robotics Res. **10**(4), 396–409 (1991)

39.14 R.G. Bonitz, T.C. Hsia: Force decomposition in cooperating manipulators using the theory of metric spaces and generalized inverses, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Diego (1994) pp. 1521–1527

39.15 D. Williams, O. Khatib: The virtual linkage: A model for internal forces in multi-grasp manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 1025–1030

39.16 K.S. Sang, R. Holmberg, O. Khatib: The augmented object model: cooperative manipulation and parallel mechanisms dynmaics, Proc. 2000 IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1995) pp. 470–475

39.17 J.T. Wen, K. Kreutz-Delgado: Motion and force control of multiple robotic manipulators, Automatica **28**(4), 729–743 (1992)

39.18 T. Yoshikawa, X.Z. Zheng: Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object, Int. J. Robotics Res. **12**, 219–230 (1993)

39.19 V. Perdereau, M. Drouin: Hybrid external control for two robot coordinated motion, Robotica **14**, 141–153 (1996)

39.20 H. Bruhm, J. Deisenroth, P. Schadler: On the design and simulation-based validation of an active compliance law for multi-arm robots, Robotics Auton. Syst. **5**, 307–321 (1989)

39.21 S.A. Schneider, R.H. Cannon Jr.: Object impedance control for cooperative manipulation: Theory and experimental results, IEEE Trans. Robotics Autom. **8**, 383–394 (1992)

39.22 R.G. Bonitz, T.C. Hsia: Internal force-based impedance control for cooperating manipulators, IEEE Trans. Robotics Autom. **12**, 78–89 (1996)

39.23 Y.-R. Hu, A.A. Goldenberg, C. Zhou: Motion and force control of coordinated robots during constrained motion tasks, Int. J. Robotics Res. **14**, 351–365 (1995)

39.24 Y.-H. Liu, S. Arimoto: Decentralized adaptive and nonadaptive position/force controllers for redundant manipulators in cooperation, Int. J. Robotics Res. **17**, 232–247 (1998)

39.25 P. Chiacchio, S. Chiaverini, B. Siciliano: Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system, ASME J. Dyn. Syst. Meas. Control **118**, 691–697 (1996)

39.26 F. Caccavale, P. Chiacchio, S. Chiaverini: Task-space regulation of cooperative manipulators, Automatica **36**, 879–887 (2000)

39.27 G.R. Luecke, K.W. Lai: A joint error-feedback approach to internal force regulation in cooperating manipulator systems, J. Robotics Syst. **14**, 631–648 (1997)

39.28 F. Caccavale, P. Chiacchio, S. Chiaverini: Stability analysis of a joint space control law for a two-manipulator system, IEEE Trans. Autom. Control **44**, 85–88 (1999)

39.29 P. Hsu: Coordinated control of multiple manipulator systems, IEEE Trans. Robotics Autom. **9**, 400–410 (1993)

39.30 F. Basile, F. Caccavale, P. Chiacchio, J. Coppola, C. Curatella: Task-oriented motion planning for multi-arm robotic systems, Robotics Comp.-Integr. Manuf. **28**, 569–582 (2012)

39.31 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Global task space manipulability ellipsoids for multiple arm systems, IEEE Trans. Robotics Autom. **7**, 678–685 (1991)

39.32 S. Lee: Dual redundant arm configuration optimization with task-oriented dual arm manipulability, IEEE Trans. Robotics Autom. **5**, 78–97 (1989)

39.33 T. Kokkinis, B. Paden: Kinetostatic performance limits of cooperating robot manipulators using force-velocity polytopes, Proc. ASME Winter Annu. Meet. Robotics Res., San Francisco (1989)

39.34 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Task space dynamic analysis of multiarm system configurations, Int. J. Robotics Res. **10**, 708–715 (1991)

39.35 D.E. Orin, S.Y. Oh: Control of force distribution in robotic mechanisms containing closed kinematic chains, Trans. ASME J. Dyn. Syst. Meas. Control **102**, 134–141 (1981)

39.36 Y.F. Zheng, J.Y.S. Luh: Optimal load distribution for two industrial robots handling a single object, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Phila. (1988) pp. 344–349

39.37 I.D. Walker, S.I. Marcus, R.A. Freeman: Distribution of dynamic loads for multiple cooperating robot manipulators, J. Robotics Syst. **6**, 35–47 (1989)

39.38 M. Uchiyama: A unified approach to load sharing, motion decomposing, and force sensing of dual arm robots, 5th Int. Symp. Robotics Res., ed. by H. Miura, S. Arimoto (1990) pp. 225–232

39.39 M.A. Unseren: A new technique for dynamic load distribution when two manipulators mutually lift a rigid object. Part 1: The proposed technique, Proc. 1st World Autom. Congr. (WAC), Maui, Vol. 2 (1994) pp. 359–365

39.40 M.A. Unseren: A new technique for dynamic load distribution when two manipulators mutually lift a rigid object. Part 2: Derivation of entire system model and control architecture, Proc. 1st World Autom. Congr. (WAC), Maui, Vol. 2 (1994) pp. 367–372

39.41 M. Uchiyama, T. Yamashita: Adaptive load sharing for hybrid controlled two cooperative manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) Sacramento (1991) pp. 986–991

39.42 M. Uchiyama, Y. Kanamori: Quadratic programming for dextrous dual-arm manipulation, Trans. IMACS/SICE Int. Symp. Robotics Mechatron. Manuf. Syst., Kobe (1993) pp. 367–372

39.43 Y.F. Zheng, M.Z. Chen: Trajectory planning for two manipulators to deform flexible beams, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 1019–1024

39.44 M.M. Svinin, M. Uchiyama: Coordinated dynamic control of a system of manipulators coupled via a flexible object, Prepr. 4th IFAC Symp. Robot Control, Capri (1994) pp. 1005–1010

39.45 T. Yukawa, M. Uchiyama, D.N. Nenchev, H. Inooka: Stability of control system in handling of a flexible object by rigid arm robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 2332–2339

39.46 M. Yamano, J.-S. Kim, A. Konno, M. Uchiyama: Cooperative control of a 3D dual-flexible-arm robot, J. Intell. Robotics Syst. **39**, 1–15 (2004)

39.47 T. Miyabe, M. Yamano, A. Konno, M. Uchiyama: An approach toward a robust object recovery with flexible manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Maui (2001) pp. 907–912

39.48 T. Miyabe, A. Konno, M. Uchiyama, M. Yamano: An approach toward an automated object retrieval operation with a two-arm flexible manipulator, Int. J. Robotics Res. **23**, 275–291 (2004)

39.49 M. Uchiyama, A. Konno: Modeling, controllability and vibration suppression of 3D flexible robots. In:

Avidbots Ex1016
Page 1037 of 2154

Robotics Research, The 7th Int. Symp, ed. by G. Giralt, G. Hirzinger (Springer, London 1996) pp. 90–99
39.50 K. Munawar, M. Uchiyama: Slip compensated manipulation with cooperating multiple robots, 36th IEEE Conf. Decis. Control, San Diego (1997)
39.51 D. Sun, J.K. Mills: Adaptive synchronized control for coordination of multirobot assembly tasks, IEEE Trans. Robotics Autom. **18**, 498–510 (2002)
39.52 A. Rodriguez-Angeles, H. Nijmeijer: Mutual synchronization of robots via estimated state feedback: a cooperative approach, IEEE Trans. Control Syst. Technol. **12**, 542–554 (2004)
39.53 K.-Y. Lian, C.-S. Chiu, P. Liu: Semi-decentralized adaptive fuzzy control for cooperative multirobot systems with H-inf motion/internal force tracking performance, IEEE Trans. Syst. Man Cybern. **32**, 269–280 (2002)
39.54 W. Gueaieb, F. Karray, S. Al-Sharhan: A robust adaptive fuzzy position/force control scheme for cooperative manipulators, IEEE Trans. Control Syst. Technol. **11**, 516–528 (2003)
39.55 W. Gueaieb, F. Karray, S. Al-Sharhan: A robust hybrid intelligent position/force control scheme for cooperative manipulators, IEEE/ASME Trans. Mechatron. **12**, 109–125 (2007)
39.56 J. Gudiño-Lau, M.A. Arteaga, L.A. Muñoz, V. Parra-Vega: On the control of cooperative robots without velocity measurements, IEEE Trans. Control Syst. Technol. **12**, 600–608 (2004)
39.57 R. Tinos, M.H. Terra, J.Y. Ishihara: Motion and force control of cooperative robotic manipulators with passive joints, IEEE Trans. Control Syst. Technol. **14**, 725–734 (2006)
39.58 H. Inoue: Computer controlled bilateral manipulator, Bulletin JSME **14**(69), 199–207 (1971)
39.59 M. Uchiyama, T. Kitano, Y. Tanno, K. Miyawaki: Cooperative multiple robots to be applied to industries, Proc. World Autom. Congr. (WAC), Montpellier (1996) pp. 759–764
39.60 B.M. Braun, G.P. Starr, J.E. Wood, R. Lumia: A framework for implementing cooperative motion on industrial controllers, IEEE Trans. Robotics Autom. **20**, 583–589 (2004)
39.61 D. Sun, J.K. Mills: Manipulating rigid payloads with multiple robots using compliant grippers, IEEE/ASME Trans. Mechatron. **7**, 23–34 (2002)
39.62 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robot hand, IEEE Trans. Robotics Autom. **5**, 151–165 (1989)
39.63 A. Jazidie, T. Tsuji, M. Nagamachi, K. Ito: Multipoint compliance control for dual-arm robots utilizing kinematic redundancy, Trans. Soc. Instr. Control Eng. **29**, 637–646 (1993)
39.64 T. Tsuji, A. Jazidie, M. Kaneko: Distributed trajectory generation for multi-arm robots via virtual force interactions, IEEE Trans. Syst. Man Cybern. **27**, 862–867 (1997)
39.65 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robotics Syst. **13**, 755–764 (1996)
39.66 J. Fink, N. Michael, S. Kim, V. Kumar: Planning and control for cooperative manipulation and transportation with aerial robots, Int. J. Robotics Res. **30**, 324–334 (2011)
39.67 J.Y.S. Luh, Y.F. Zheng: Constrained relations between two coordinated industrial robots for motion control, Int. J. Robotics Res. **6**, 60–70 (1987)
39.68 A.J. Koivo, M.A. Unseren: Reduced order model and decoupled control architecture for two manipulators holding a rigid object, ASME J. Dyn. Syst. Meas. Control **113**, 646–654 (1991)
39.69 M.A. Unseren: Rigid body dynamics and decoupled control architecture for two strongly interacting manipulators manipulators, Robotica **9**, 421–430 (1991)
39.70 J. Duffy: The fallacy of modern hybrid control theory that is based on *Orthogonal Complements* of twist and wrench spaces, J. Robotics Syst. **7**, 139–144 (1990)
39.71 K.L. Doty, C. Melchiorri, C. Bonivento: A theory of generalized inverses applied to robotics, Int. J. Robotics Res. **12**, 1–19 (1993)
39.72 O. Khatib: Object manipulation in a multi-effector robot system. In: *Robotics Research*, Vol. 4, ed. by R. Bolles, B. Roth (MIT Press, Cambridge 1988) pp. 137–144
39.73 O. Khatib: Inertial properties in robotic manipulation: An object level framework, Int. J. Robotics Res. **13**, 19–36 (1995)
39.74 F. Caccavale, P. Chiacchio, A. Marino, L. Villani: Six-DOF impedance control of dual-arm cooperative manipulators, IEEE/ASME Trans. Mechatron. **13**, 576–586 (2008)
39.75 T.G. Sugar, V. Kumar: Control of cooperating mobile manipulators, IEEE Trans. Robotics Autom. **18**, 94–103 (2002)
39.76 C.P. Tang, R.M. Bhatt, M. Abou-Samah, V. Krovi: Screw-theoretic analysis framework for cooperative payload transport by mobile manipulator collectives, IEEE/ASME Trans. Mechatron. **11**, 169–178 (2006)
39.77 H. Bai, J.T. Wen: Cooperative load transport: A formation-control perspective, IEEE Trans. Robotics **26**, 742–750 (2010)

Avidbots Ex1016
Page 1038 of 2154

Multimedia Contents

40. Mobility and Manipulation

Oliver Brock, Jaeheung Park, Marc Toussaint

Mobile manipulation requires the integration of methodologies from all aspects of robotics. Instead of tackling each aspect in isolation, mobile manipulation research exploits their interdependence to solve challenging problems. As a result, novel views of long-standing problems emerge. In this chapter, we present these emerging views in the areas of grasping, control, motion generation, learning, and perception. All of these areas must address the shared challenges of high-dimensionality, uncertainty, and task variability. The section on grasping and manipulation describes a trend towards actively leveraging contact and physical and dynamic interactions between hand, object, and environment. Research in control addresses the challenges of appropriately coupling mobility and manipulation. The field of motion generation increasingly blurs the boundaries between control and planning, leading to task-consistent motion in high-dimensional configuration spaces, even in dynamic and partially unknown environments. A key challenge of learning for mobile manipulation consists of identifying the appropriate priors, and we survey recent learning approaches to perception, grasping, motion, and manipulation. Finally, a discussion of promising methods in perception shows how concepts and methods from navigation and active perception are applied.

40.1 **Grasping and Manipulation** 1009
40.1.1 Problem Description 1009
40.1.2 Evaluation of the State of the Art 1010
40.1.3 Toward Robust Grasping and Manipulation 1013
40.2 **Control** 1013
40.2.1 Problem Description 1014
40.2.2 Evaluation of the State of the Art 1014
40.2.3 Toward Control in Mobile Manipulation 1016
40.3 **Motion Generation** 1017
40.3.1 Problem Description 1017
40.3.2 Evaluation of the State of the Art 1018
40.3.3 Toward Motion Generation for Mobile Manipulation 1020
40.4 **Learning** 1021
40.4.1 Problem Description 1021
40.4.2 Evaluation of the State of the Art 1023
40.4.3 Toward Learning in Mobile Manipulation 1025
40.5 **Perception** 1025
40.5.1 Problem Description 1026
40.5.2 Evaluation of the State of the Art 1026
40.5.3 Toward Robust Perception 1028
40.6 **Conclusions and Further Reading** 1029
Video-References 1029
References 1030

What exactly is mobile manipulation? To find an answer to this question, we must consider the historical development of this research area. The term mobile manipulation was coined in the late 1980s to early 1990s, when research labs began to mount robot manipulators on mobile platforms [40.1, 2]. Back then, the term was intended to capture just that: research was conducted on an experimental platform that combines capabilities in mobility and manipulation. A series of mobile manipulation platforms is shown in Fig. 40.1.

With the introduction of such platforms it soon became apparent that combining mobility with manipulation was a game changer. Mobility enables manipulators to leave the lab. They now have to face the complexities of the real world. And in the real world, much of the research that had been successful in the carefully

controlled environments in research labs proved to be brittle or inadequate. This traditional research often relied on strong assumptions about the environment to be successful – an untenable assumption in unpredictable, nonstationary everyday environments.

In the context of mobile manipulation research, controlled environments, such as factory floors and specific experimental setups, are often referred to as *structured environments*. In contrast, *unstructured environments* are environments that have not been modified specifically to facilitate the execution of a task by a robot. These two types of environments are opposites on a continuous scale with many possible intermediates. And, of course, structured environments are still laden with uncertainty just the same as unstructured environments still contain a significant structure that can be exploited by the robot.

In unstructured environments, it becomes a necessity for the robot to perform a wide variety of tasks, instead of a single specific task. Any specific task, such as retrieving a book from the library, might require the robot to solve additional challenges, such as opening doors, operating elevators, moving a chair out of the way to approach the shelf, or asking for the location of the stacks. Today, no robotic system possesses such capabilities.

Researchers in mobile manipulation have learned that the progress achieved in traditional research areas of robotics cannot simply be combined to produce the level of competency they aim for. This insight motivated the founding of mobile manipulation as a research area. The research in this area seeks to develop robotic systems capable of autonomous task execution in unstructured or minimally structured environments. Clearly, the transition from single-purpose lab demonstration – a common practice in robotics – toward such mobile manipulation systems is a gradual one. Mobile manipulation research therefore gradually increases the autonomy, robustness, and task generality of robotic systems, while at the same time gradually reducing the dependence on prior information about the environment and the task.

The objective of mobile manipulation research is to:

1. Maximize task generality of autonomous robotic systems, while at the same time
2. Minimizing the dependence on task-specific, hard-coded, or narrowly-relevant information.

Some people argue that this is exactly what main stream robotics is doing and what most of this Handbook is about. And they might be right. But over the last 20 years, the mobile manipulation community, forced by failures of robotic systems in the real world, has begun to explore alternatives and extensions to the body of robotics knowledge collected in this tome. It is these alternatives or extensions that this chapter is about. In each of the subsequent sections, our goal is to showcase work in mobile manipulation that opens up possible novel directions for future advances toward robots operating in real-world environments. For the areas of grasping and manipulation, mobility, control, motion generation, learning, and perception, we will analyze how the main challenges of the field change when systems are operating in the real world. We will show how each of these fields has developed and advanced to increasingly address the challenges of unstructured environments.

Before we embark on this journey, it is worthwhile to agree on the challenges that are in fact encountered when robots are supposed to perform a variety of tasks in an unconstrained and uncontrolled envi-



Fig.40.1a–d Some exemplary mobile manipulation platforms developed throughout the decades **(a)** Herbert (MIT, 1985), **(b)** Stanford assistent mobile manipulator (1995), **(c)** PR2 (Willow Garage, 2009), **(d)** Herb (CMU, 2013)

ronment. These challenges are common to all of the subdisciplines we discuss in the following sections.

First, the necessity to perform a variety of tasks, instead of a single one, requires that the robot be equipped with general means of mobility, manipulation, and perception. Its capabilities must be the union of the minimum capabilities required for each task. This entails the need for versatile sensing and dexterous manipulation. This, in turn, implies that the sensor input of the robot is high dimensional and so is the configuration space of the robot. The first challenge therefore consists of dealing with the *high dimensionality* and the inherent complexity of the state spaces required for task generality. To make matters worse, task generality most often requires system dynamics to be hybrid, i. e., exhibit discrete as well as continuous behavior.

Second, by the definition of mobile manipulation, the robot cannot rely on a complete, continuously, and globally updated world model. While some part of the robotics community attempts to structure the world to become such a world model – by using RFID tags or by placing sensors everywhere – many in the mobile manipulation community believe it to be more effective and more insightful to make robots smarter. Not having a complete world model means that the robot has to use perception to acquire relevant information about the world; it means that the robot has to address the uncertainty in its sensing, but also the uncertainty in its world model which might change unbeknownst to the robot. The second challenge therefore consists of appropriately handling the *uncertainty* inherent to sensing and actuation and the uncertainty caused by a dynamic world.

Third, in the real world, objects exhibit large variability from the perspective of their appearance, even when they perform exactly the same function. This means that there is a layer of indirection through appearance when a robot wants to operate autonomously in the real world. There are so many different kinds of door handles, they differ significantly in appearance, but perform the same function. The challenge therefore is to cope with the *variability* in the world by revealing the functional principles underlying the world's appearance. A natural way of addressing variability are skills that generalize to novel situations.

As we will see in subsequent sections, attempts to address these challenges by the robotics community have produced innovative and successful concepts, algorithms, mechanisms, and integrated systems. Some of them were not covered in previous chapters, as they do not (yet?) belong to the canon of robotic wisdom. Maybe this chapter can contribute to changing this.

40.1 Grasping and Manipulation

Robots accomplish tasks by grasping, manipulating, and placing objects. Several chapters in this Handbook are concerned with associated mechanisms, methods, and concepts: Chapter 19 discusses robot hands, the most versatile type of end-effector for grasping and manipulation. Chapter 37 discusses manipulation in general and Chap. 38 presents the state of the art in grasping. Important for grasping and manipulation are also force and tactile sensors, which are discussed in Chap. 28. While these four chapters describe the well-established foundation of grasping and manipulation, in this chapter we want to take a higher level perspective, assess the suitability of these approaches in the context of mobile manipulation, and speculate about possible novel approaches.

Both grasping and manipulation exemplify the challenges for mobile manipulation we laid out above: uncertainty, high dimensionality, and task variability. Uncertainty is relevant as minor inaccuracies in the object model, minor errors in motion execution, and small differences in positioning relative to the object can cause grasp failure. The high dimensionality of grasping results from the many degrees of freedom of the hand as well as from those of the objects in the environment. Finally, grasping and manipulation occur in a large number of variations, each involving vastly different interactions with objects. It is therefore not surprising that decades of research on grasping and manipulation have made significant progress but so far were unable to create skills sufficiently robust for autonomous operation in the real world.

In this section, we describe a recent development in the field of grasping and manipulation away from the foundational techniques laid out elsewhere in this Handbook toward novel approaches that seem better suited for addressing the manipulation challenges in mobile manipulation.

40.1.1 Problem Description

First, a word on defining the problem. Traditionally, grasping and manipulation are viewed as two distinct problems. But there is signifiifant ambiguity with respect to these terms in the literature [40.3]. In this section, we limit ourselves to grasping and manipulation with robot hands. And in this context, we define the following terms:

Grasping refers to gaining reliable control over the extrinsic degrees of freedom of an object through the robot's degrees of freedom, i. e., the robot grasps an object so that it can change the object's extrinsic degrees of freedom by changing its own, excluding the degrees of freedom of the hand, which remain static once the grasp is achieved. To perform grasping, the overall degrees of freedom of the robot have traditionally been divided into two parts: those required to impart forces on an object (usually addressed using grasp planning approaches) and the remaining ones that can be used to reposition the object, once a grasp is obtained (general motion and manipulation planning). Note, however, that in contrast to this traditional decomposition, coordination of arm and hand degrees of freedom are usually required during grasping.

By *dexterous grasping* we refer to grasping while optimizing the grasp posture for (a) robustness and (b) suitability for the execution of a specific task. In the relevant literature, the overall degrees of freedom are still most commonly divided into those responsible for the grasp and those able to change the pose of the grasped object.

By *dexterous in-hand manipulation*, we refer to the ability to actuate the extrinsic degrees of freedom of the object using all degrees of freedom of the arm and the hand so that the object moves relative to a reference frame placed at the hand's wrist [40.4].

Classical approaches to all three of these problems rely on precise models of the environment and of the hand, carefully reason about contact points and states, and attempt to determine and precisely execute a detailed plan. We would like to point to the emergence of a novel approach to all three of these problems. Since this novel approach, however, has been mainly explored in the context of grasping at this early point in time, we will focus our discussion on grasping. But we will also make the point that in this new way of thinking about grasping and manipulation both of these problems become similar, if not identical.

40.1.2 Evaluation of the State of the Art

The earliest and still most fundamental considerations of grasping are based on the notions of force and form closure (Chap. 38).

Force and Form Closure

These notions express the effect of a set of disembodied contact points, i. e., ignoring the body on which that contact point lies, such as the finger, on the ability of an object to move. They reflect a static view of grasping, in which the physical interactions that will invariably occur during the grasp are not considered.

This line of research continues to be active and successful, as evidenced by a large number of sophisticated and capable grasp planners and simulators [40.5]. The resulting grasps, however, often do not result in successful real-world grasps. These failures are a consequence of model uncertainty and motion execution noise. The assumptions made during planning, i. e., that the disembodied contact points can be realized precisely by the hand, more often than not do not hold.

Interactions Between Hand and Object

There is a very simple strategy to overcome many of the theoretical failures of the approaches based on precise models and the notion of force closure – and this strategy is used tacitly by all grasping research: just keep closing the hand, even if you think you have attained the right configuration for your contact points. This strategy effectively leverages mechanical compliance of the hand to adapt the hand's configuration to the object's shape. This adaptation of shape has several positive effects for grasping:

1. Sensing and actuation uncertainty are compensated,
2. Large surface areas are established, and
3. Grasping forces are distributed and balanced.

These effects lead to an increase in grasp success and grasp quality. It is fair to say that this is a standard trick in the grasp planning community.

This insight is also strongly driving contemporary hand design. To maximize the effect of shape adaptation, *Rodriguez* and *Mason* [40.6] optimize finger shape to yield similar contact configuration irrespective of the size of the object. A different way of increasing shape adaptability is to include compliant components in the hand design [40.7]. The shape deposition manufacturing (SDM) hand [40.8], the Velo gripper [40.9], the i-HY hand [40.10], and the Pisa/IIT SoftHand [40.11] achieve shape adaptability through underactuation by coupling degrees of freedom of the hand, adapting the shape of the hand to the object while equalizing contact forces. An extreme case of this is the positive pressure gripper [40.12], which presses a bag filled with granular material onto the object to be grasped. The bag adapts to the shape of the object. When the air is evacuated from the bag, the granular material jams, forming a gripper perfectly adapted to the shape of the object.

From this brief analysis it is fair to say that the shape adaptability between hand and object is routinely leveraged in grasp planning, simply by *closing the hand* until a certain grasping force is reached. Also, shape adaptability has emerged as an important design criterion for hands, as it very obviously improves grasping performance.

Interactions Between Hand, Object, and Environment

Features in the environment, provided by other objects or support surfaces, for example, may constrain the motion of hand and object. This is most evident for support surfaces, such as tables and floors. These constraints, when used properly, can aid grasping. Furthermore, it might be the case that the necessary perceptual information for leveraging such constraints is often easier to obtain than the information required for the successful execution of an unconstrained grasp.

Recent research in robotic grasping leverages environmental constraints in the suggested manner, e.g., to position the hand relative to the object [40.13], to cage objects [40.13, 14], or to fixate an object during planar sliding [40.13, 15]. Some pregrasp manipulation relies on environmental constraints to improve grasp success. For example, *Chang* et al. [40.16] rotate pan handles into a specific orientation prior to grasping by exploiting the pan's friction and remote center of mass.

But, in addition to external contacts, environmental constraints can also be created by exploiting gravity, inertia, or dynamic motion of the arm and hand. In this way, in-hand manipulation is accomplished by using what has been called extrinsic dexterity [40.17].

All of the aforementioned grasp strategies rely on multiple interactions between hand, object, and environment prior to attaining the final grasp posture. These phases often are designed to reduce uncertainties in specific variables relevant to grasp success.

The idea of environmental constraints appears in early work by *Mason* et al. [40.18, 19]. Here, the intrinsic mechanics of the task environment are exploited to eliminate uncertainty and to achieve robustness.

The study of environmental constraint exploitation by humans so far has been limited to replicating instances of observed behavior on robots [40.16, 20]. For example, *Kaneko* et al. [40.20] extracted a set of grasping strategies from observations of a human subject. These strategies include interactions with environmental constraints.

We believe that this recent trend toward the exploitation of environmental constraints represents an important opportunity to improve robotic grasping capabilities. To take full advantage of this opportunity, we should understand the strategies humans employ, transfer them to robotic systems, and develop robotic hands tailored to this exploitation [40.21]. This will be discussed in the next section.

Insights from Human Grasping

Interestingly, the study of human grasping paralleled the development in the robotics community. Early studies of human grasping followed the static view captured by force closure concepts. This is reflected in grasp taxonomies, classifying grasp according to the final hand posture attained after the grasp process is completed [40.22, 23].

Even the early work on postural synergies, which has had a profound impact on robotics, initially only considered synergies of static grasp postures [40.24]. These studies do not capture the dynamic processes and the exploitation of environmental constraints we believe to be crucial for robust grasping. Nevertheless, these studies of human grasping have led to significant advances in robotic grasping [40.11].

The nature of hand–object interaction has also been studied in humans. The effect of shape adaptability is well known for human hands and studies have elucidated the degree to which humans vary their behavior to take advantage of it. *Christopoulos* and *Schrater* [40.25] showed that humans react to pose uncertainty of a cylinder and orient their hand to align it, presumably to be able to maximize the benefits of shape adaptability. However, other experiments may point at the fact that humans also rely on more complex interactions with the environment for grasping under difficult conditions. When the vision of humans is impaired, they fail more often at first grasp attempts of isolated (environmental-constraint free) objects [40.26]. The degree of the demonstrated effect seems surprising. We believe that in this specific experiment, it is due to the lack of environmental constraints exploitable for grasping.

More recently, studies of human grasping have shown the deliberate use of environmental constraints during grasping. For example, humans increase their use of environmental constraints when their vision is impaired [40.21]. This seems to indicate that the use of environmental constraints will play a crucial role in achieving robust and task-general grasping and manipulation in robotics.

Robot Hands

Many highly capable robotic hands exist. A historical overview, collecting robotic hands from over five decades, was compiled by *Controzzi* et al. [40.27]. An analysis of robot hand designs with respect to grasping capabilities was recently presented by *Grebenstein* [40.28]. As the notion of compliance is central to our hand design, we will limit our discussion to hands designs that deliberately include this concept.

We distinguish two main approaches for designing compliant hands. Compliance can be achieved using active control, and implemented on a fully actuated or even hyper-actuated systems, where every degree of freedom can be controlled. An impressive exam-

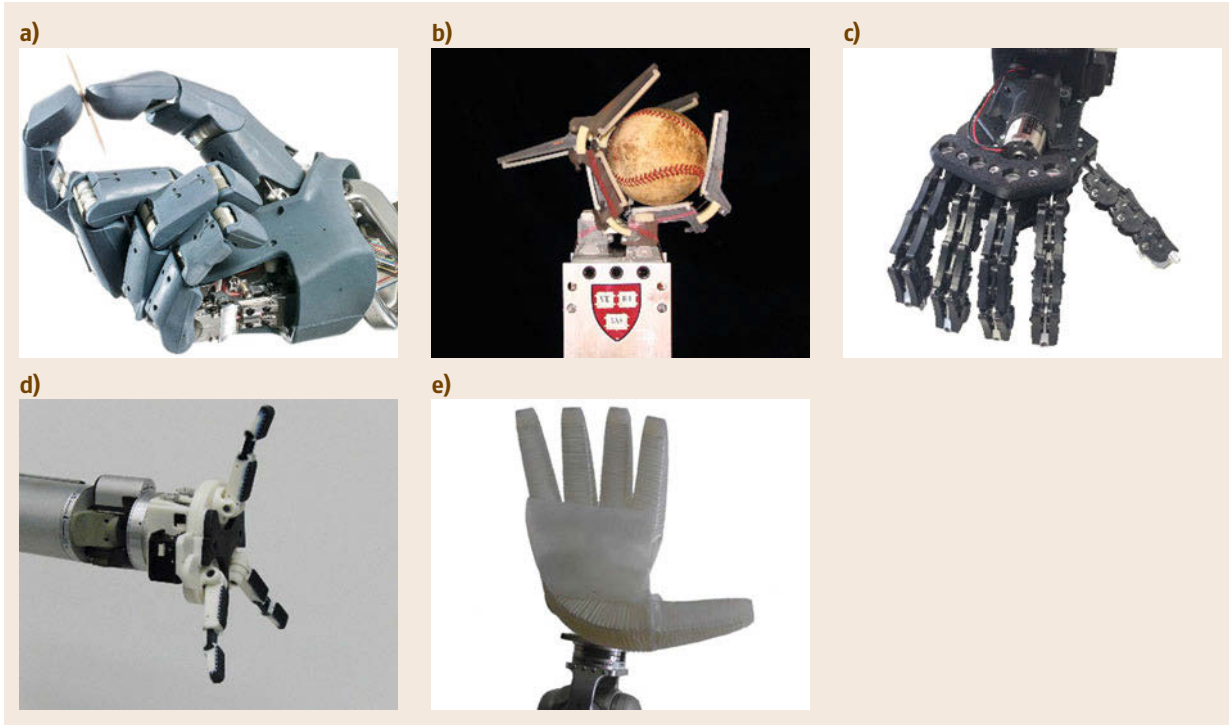


Fig.40.2a–e Various hand designs leveraging compliance to improve performance. **(a)** Awiwi hand (DLR), **(b)** SDM hand (Havard), **(c)** Pisa-IIT hand, **(d)** i-HY hand (iRobot, Havard, Yale), **(e)** RBO hand z (TU Berlin)

ple of this type of hand is the Awiwi hand [40.28], the ShadowRobot Shadow Dexterous Hand, and the SimLab Allegro Hand [40.29]. These hands achieve dexterity through accurate control, which comes at the price of mechanical complexity, making them difficult and costly to build and prone to failure.

An alternative approach is to make hands compliant by including elastic or flexible materials (passive compliance). Building a passively compliant joint is much cheaper than an actively controlled one, in terms of costs, volume and system complexity. Passive compliance can easily absorb impact forces – a desirable property for an end-effector designed to establish contact with the world. The cost of adding additional (passive) degrees of freedom is low, compared to actively compliant hands. The resulting ability to passively adapt to the shape of an object greatly enhances grasp success and grasp quality. At the same time, the hand can be underactuated, effectively offloading control to the physical material.

A pioneering work in grasping with passive compliance was the soft gripper by *Hirose* and *Umetani* [40.7]. Recently, a whole range of grippers and hands were built using passive compliance, such as the FRH-4 hand [40.30], the SDM hand and its successor [40.10, 31, 32], the starfish gripper [40.33], the THE Second Hand and the Pisa-IIT Soft Hand [40.34], the Positive Pressure Gripper [40.12], the RBO Hands [40.13, 35], and the Velo Gripper [40.9]. A different source of inspiration was taken by *Giannaccini* et al. [40.36], who built an octopus-inspired compliant gripper. Some of these hands are shown in Fig. 40.2.

The practical realization of underactuated hands is matched by theoretical approaches to analyze and evaluate their dexterity [40.37, 38]. However, these approaches require accurate knowledge of grasp posture, contact point locations, and contact forces. Given today's sensor technologies, this information is difficult to obtain in physical implementations.

The inclusion of compliance into the design of robotic hands has led to significant improvements in power-grasping of objects. Given the insights obtained from human grasping (see the previous section), it is plausible that these improvements are the result of these hands' capabilities to engage in beneficial contact interactions with the environment. While these benefits have been demonstrated extensively in the context of power grasping, little work has examined the beneficial effects of compliance and underactuation on the dexterity of robotic hands [40.35].

Avidbots Ex1016
Page 1044 of 2154

40.1.3 Toward Robust Grasping and Manipulation

The key to robust grasping and manipulation – an important prerequisite for mobile manipulation – appears to be the deliberate and purposeful use of interactions between hand, object, and environment. This is supported by evidence from human grasping as well as from the design features of successful robotic hands. If this claim is true, it fundamentally changes the game in grasping and manipulation. The notion of a specific contact point configuration, which has for the last 50 years been of pivotal importance, is superseded by deliberate constraint exploitations. The traditional, static view of grasping is replaced with a dynamic view that emphasizes motion of hand, object, and environment in contact with each other. Rather than placing fingers carefully at exactly the right place, this new way of thinking requires fingers to slide across surfaces, to comply to object shapes, and to balance and maintain contact forces through intelligent hardware design.

Especially in hardware design, there has been significant progress. The newest generation of hands are built for the exploitation of constraints: without sensing and explicit control, they support shape adaptation and compliant contact maintenance. One of the future challenges will be the development of grasp planning methods and perceptual capabilities to leverage the advantages afforded by these novel hands.

40.2 Control

Mobile manipulators are robotic systems with capabilities for both manipulation and mobility. A typical setup of mobile manipulator, therefore, consists of manipulator and mobile robot. Traditionally, mobile manipulators were implemented as wheeled mobile robots, providing mobility on two-dimensional flat surfaces. Mobility, however, can be provided by other types of robots depending on the operating environment. Legged robot systems operate well in uneven or rough terrain. Aerial or underwater robots allow operation in environments with totally different characteristics.

Controlling such mobile manipulators involves control for the subsystems. The control issues of manipulators are explained and discussed earlier in Chaps. 8 and 9. Modeling and control of various types of mobile platforms will be introduced later in detail in Chaps. from 48 to 52 in Part E. In these chapters, we would like to discuss unique control problems in mobile manipulation, which arise when the manipulator and mobile platform operate simultaneously.

Control of mobile manipulators also has to address the three challenges we stated initially: high dimensionality, uncertainty, and task variabilty. The mobile manipulator was first implemented to provide mobility to the manipulator. One of the main purposes is to increase the manipulation workspace of the robot (Fig. 40.3). This mobile platform naturally adds more degrees of freedom to the system by providing mobility. The additional degrees of freedom increase the dimensionality of the control problem. The robot be-

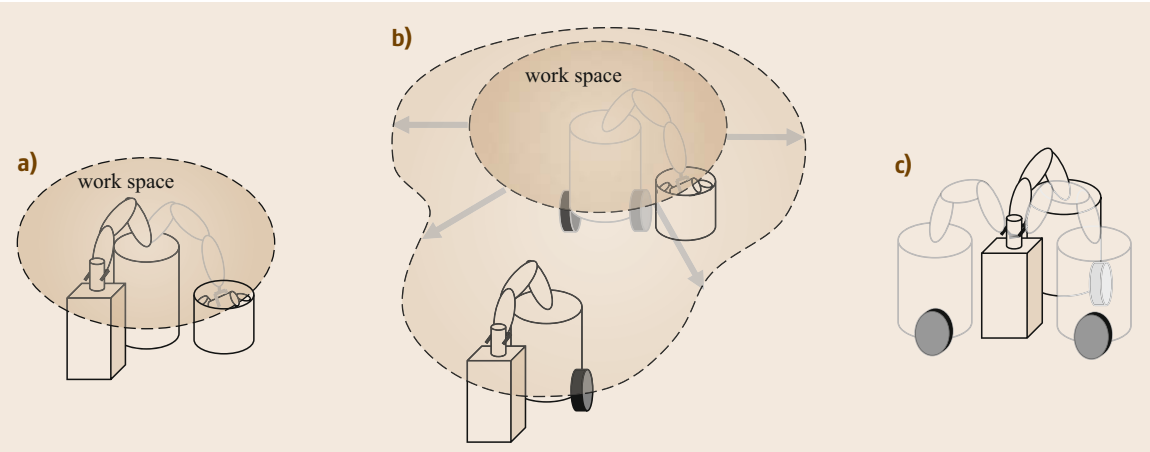


Fig.40.3a–c Increased workspace and redundancy of mobile manipulators. **(a)** A manipulator on a fixed base; **(b)** increased work space of a mobile manipulator; **(c)** redundancy due to the added mobility

comes a redundant system because the specified task to the robot is still the same, while the overall degrees of freedom have been increased. That is, there are infinite number of configurations that can execute the same task.

Second, uncertainty becomes severe in control of mobile manipulators due to the mobile platform. The uncertainty is larger when controlling a mobile platform compared to control of a manipulator. The reasons for the higher uncertainty are imperfect information or model about the operating environment, such as ground and underwater. Additionally mobile platforms with only internal sensors do not have access to perfect state information because drift raises uncertainty while moving. However, mobile manipulators need to solve the same manipulation tasks as stationary manipulators. They have to overcome this uncertainty to operate reliably.

Third, the variability of the mobile platform exists depending on the operating environment, although the high dimensionality and uncertainty are the common factors across the various mobile platforms. The different issues due to various mobile platforms are also to be discussed later in this chapter.

40.2.1 Problem Description

Control of mobile manipulators must provide capabilities for manipulation and mobility. Often times, these two are treated completely separate. The robot moves to a certain location using its mobile platform. Then, the manipulator performs specific tasks without moving the platform. In this approach, the mobile platform considers the manipulator as a static load. The mobile platform is then considered as a fixed base for the manipulator. There is no major difficulty implementing this approach, unless manipulator or mobile platform is unable to hold its position steadily.

The main problem to be discussed in this chapter arises when both manipulator and mobile base operate simultaneously. Although the previously mentioned approach of a separately controlling manipulator and a platform increases the workspace of the robot, it does not exploit the full capabilities of mobile platforms. Instead, the advantage of mobile manipulator can be maximized when the manipulator operates while the bases moves in coordination. This kind of operation will not only reduce the total operation time but will also increase task variability.

In this chapter, therefore, we will discuss how to control mobile manipulators by operating both subsystems at the same time. During this operation, the whole system becomes redundant due to the added degrees of freedom (DOF) from the mobile platform. How to exploit this increased number of DOF is one important problem in mobile manipulation. Secondly, the increased uncertainty due to the mobile platform requires more robust control strategies than that only for manipulator. Finally, various issues for different types of mobile platforms will be discussed.

40.2.2 Evaluation of the State of the Art

First mobile manipulators were implemented as wheeled robots and wheels are still the most popular tool to enable a manipulator to move. Most of the issues that have been dealt with in wheeled mobile manipulator can be shared with other types of mobile platforms. Therefore, we start discussing wheeled mobile platforms in this chapter and discuss issues specific to other types of mobile platforms afterward.

Redundancy Resolution and Control

One of the main issues in controlling mobile manipulators has been how to use the additional DOFs to execute manipulation tasks in a stable way. An early study [40.39] investigated how to obtain solutions to the inverse kinematics problem that prevent the robot from tipping over.

Other approaches exploited the increased number of DOFs to avoid obstacles, increase manipulability, handle singularities, or achieve lower priority tasks. In [40.40], a preferred region to maximize the manipulability of the robotic arm is defined and the nonholonomic mobile platform is controlled to be within the region. The same concept of a preferred region is also applied to the force control of a mobile manipulator in [40.41]. These approaches are typical examples for approaches that coordinate with the manipulator and base motion, but control the subsystems separately.

The redundancy of mobile manipulators allows dealing with multiple tasks simultaneously. In the configuration approach [40.42, 43], additional tasks or specifications are defined to be simultaneously controlled with the end-effector. To obtain solutions online, a weighted damped least-squares approach was presented in [40.44]. The configuration approach is further developed to be used both for holonomic and nonholonomic bases in [40.45].

In [40.46], the unified control of the arm and holonomic base was achieved by applying the operational space control framework to the mobile manipulator. The coordination of the arm and base can be implemented in the null-space of the task control, which does not affect the control of the end-effector. Later, this approach was extended to deal with nonholonomic bases in [40.47]. It uses a complete model of the system with nonholonomic constraints to devise one unified con-

troller for the whole system. The redundancy was used to deal with internal and external constraints. Another model-based control is derived in [40.48] for a car-like and differentially-driven system.

Similarly, an approach of exploiting redundancy for avoiding singularities and maximizing manipulability is proposed in [40.49] using an event-based planner and a nonlinear feedback controller with the dynamic model of the robot.

As can be seen from the above literature, the issue of redundancy resolution in mobile manipulator has been often dealt with as an extension from the control of redundant systems but with the special case of nonholonomic constraints.

Uncertainty Due to Mobile Platform

Another main problem arising when controlling mobile manipulators, is the additional uncertainty due to the mobile platform. This is because the mobile platform in real world moves on the imperfect ground, which creates a large uncertainty on the manipulation although it can be negligible for navigation. Early work in the nineties studied how to control the manipulator when the base is under disturbance due to the ground condition in [40.50]. The disturbance was assumed to be unknown and considered as rotational motion only.

Also, model-based approaches are not always suited for the control of mobile manipulator because accurate models of mobile platforms are harder to obtain than models of manipulators. The extended Jacobian transpose algorithm is proposed in [40.51] for controlling the manipulator to overcome the excessive end-effector error or instability due to unmodeled vehicle dynamics. Similarly, since precise modeling of the interaction is difficult, decentralized control is proposed in [40.52]. It regards the interaction force from each system as unknown disturbance.

In [40.53], the control system is divided into mobile platform and manipulator, and two low-level controllers are used to control each system. The dynamic interaction force from the base is modeled as an unknown disturbance in manipulator control, and an adaptive controller was designed to deal with this disturbance. A redundancy resolution scheme avoids singularities. On the other hand, the effects of dynamic interactions between manipulator and wheeled base on the performance of the end-effector task are studied in [40.54].

In another aspect, there have been approaches without modeling interaction or dynamics of each system. A robust damping controller is proposed in [40.55] for the motion control subject to kinematic constraints but without any knowledge of dynamic parameters.

Neural network-based control also has been proposed in this respect. In [40.56], a neural network-based joint space control is developed for a mobile manipulator with unknown dynamic model and disturbance. Two neural network controllers are applied to manipulator and vehicle, respectively. The radial basis function network with weight adaptation is used in [40.57] for online estimation of nonlinear dynamics. The algorithm is applied to a simulation of a two-link manipulator on a mobile platform. Adaptive robust motion/force control has been developed for both holonomic and nonholonomic constrained mobile manipulators in [40.58]. It ensures stability and the boundedness of tracking errors in the presence of uncertainties in dynamic parameters and disturbances.

Sliding mode control and adaptive neural network control are combined together in [40.59]. Multilayered perceptrons are applied to estimate the dynamic model as one system, and the adaptive control is designed in the task space control. A sliding mode approach with neural network-based control is proposed in [40.60] for omnidirectional wheeled mobile manipulators.

Control Issues on Various Types of Mobile Platform

Although wheeled vehicles are one of the most general types of platform to provide mobility, there is an increasing tendency of using other types (Fig. 40.4).

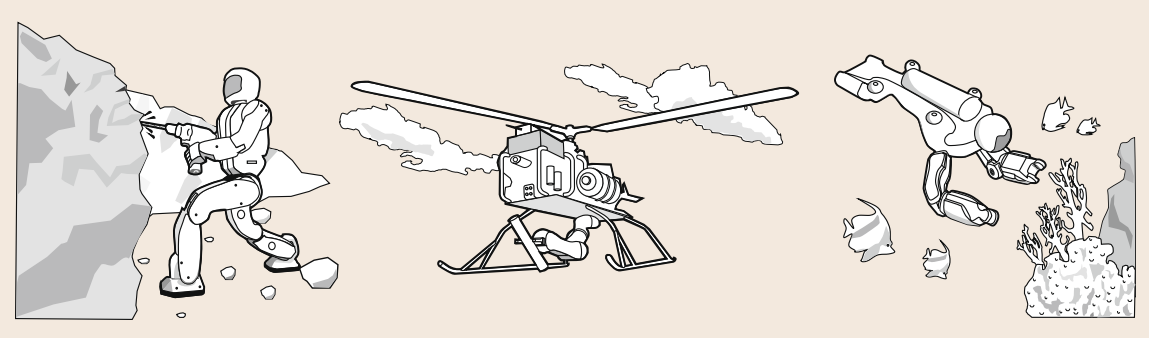


Fig. 40.4 Mobile manipulators in various environments

Legged locomotion systems like humanoids and quadrupeds are becoming popular. The interesting new aspect of these systems is that the platform for locomotion can actively participate in manipulation. That is, the legs of the humanoid system can provide not only locomotion but also assist in manipulation. This is especially enabled by using the whole-body control framework [40.61–64]. This control framework considers the robotic system as a whole and utilize all the joints for both manipulation and locomotion. This is especially justified in legged robots because the mobile platform and manipulator use the same type of actuator. Other platforms use different types of actuation for base and manipulator.

Also, quadruped robots such as Robosimian from NASA JPL uses the leg parts to increase the workspace and dexterity of the manipulation. The legged system can provide more capability of manipulation than wheeled mobile platform because it can provide the full three-dimensional (3-D) motion of the base part such that the manipulation part can have much larger manipulation capability. The whole-body control framework is also applied to a quadrupedal robot in [40.65]. The robot does not yet have manipulator in this system, but the control framework can support the manipulation control if needed.

Mobile manipulators operating underwater are becoming more prevalent. The main difference to wheeled mobile platform is the effect of hydrodynamic forces during motion. An efficient dynamic simulation algorithm is developed for underwater mobile manipulator in [40.66], where various hydrodynamic forces can be modeled and incorporated. In [40.67], the hydrodynamic effect between the manipulator and vehicle is modeled and compensated. The demonstrated results improved performance in controlling both vehicle and manipulator. The underwater mobile manipulator is modeled using Kane's method in [40.68], which also incorporates the major hydrodynamic forces such as added mass, profile drag, fluid acceleration, and buoyancy.

Underwater mobile manipulators also have redundant degrees of freedom. In [40.69], redundancy created by the underwater vehicle is utilized to achieve secondary tasks, such as reducing energy consumption and maximizing dexterity, while executing the main task at the end-effector. A task-priority redundancy resolution technique is applied for this purpose. The restoring moment is minimized in [40.70] in redundancy resolution so that the performance of the coordinated motion control is improved.

Similar to other mobile platforms, the uncertainty due to the underwater vehicle is one of the major difficulties. To deal with this issue, an adaptive control approach is proposed in [40.71] that is robust in the presence of uncertainties in the vehicle and environment. An iterative learning algorithm is developed for the hydrodynamic effects in [40.72]. An adaptive tracking control is proposed in [40.73] that keeps the advantage of model-based control and has a modular structure. An observer–controller strategy deals with the difficulty of obtaining precise velocity measurements in [40.74]. This significantly improves the chattering of the output actuator caused by noise and quantization.

Aerial mobile manipulation has been actively investigated in recent years. Aerial mobile manipulation has different characteristics in another sense: stability and load capability. The dynamics of the manipulator or interactions with the environment can create stability problem in aerial mobile manipulation. Also, interaction forces and load are limited due to the capability of lift provided by the aerial platform.

An aerial mobile manipulator system with three 2-DOF arms is developed in [40.75]. The interaction between the manipulators and quadrotors is modeled so that the reaction forces and torques during flight and manipulation are compensated for stable flight.

A Cartesian impedance control is implemented in [40.76] and [40.77], while the task redundancy due to the added DOF from the aerial vehicle is used for secondary tasks in [40.76]. Hybrid force and motion control of tool operation in quadrotors are proposed in [40.78]. The dynamics of quadrotors are transformed into the tool-tip position and decomposed into the tangential and normal directions with respect to the contact surface. Then, a stabilizing controller is designed for the hybrid motion and force control.

40.2.3 Toward Control in Mobile Manipulation

Control of mobile manipulators is one of the most investigated topics since the beginning of the concept of mobile manipulation. Although the issues discussed so far in this chapter are still very important and need to be further investigated, interaction with humans and the environment is another important aspect to handle to enable mobile manipulators to come into our daily life. In this respect, force control at the end-effector of the mobile manipulator has been studied in [40.79, 80]. The issue of contact transition has been studied for nonholonomic mobile manipulators in [40.81]. A control strategy is proposed to deal with impulsive contact force on the mobile manipulator in [40.82].

On the other hand, the use of compliant actuators such as series elastic actuators (SEA) and variable stiffness actuators (VSAs) is expected to become more

popular. The topic of compliant actuators is covered extensively Chap. 21. The SEA has been developed for the purpose of safety and energy efficiency in the compromise of precise control. The VSA can overcome the disadvantage of SEA with additional actuation for changing stiffness.

Despite the complexity of implementing these actuators in terms of both hardware and algorithms, they are expected to be used more because there is an increasing need for safe robot–human or robot–environment interaction. The application of compliant actuators to mobile manipulators would provide new challenges to the control issues of dealing with uncertainty and stability in addition to the precision of the manipulator itself. The compliance on the joints will provide larger uncertainty in the manipulator. This uncertainty affects not only the control of the manipulator but also the control of the mobile platform. The effect on the mobile base will be the stability problem in addition to the precision in control. Nevertheless, the introduction of soft/compliant actuators will be one of the important future directions to enable safe interaction in mobile manipulation.

40.3 Motion Generation

Methods to generate the motion of robots are fundamental to robotics. They are treated extensively in this Handbook (Chaps. 7, 8 as well as the entire Part E). As in the previous sections of this chapter, we will consider the state of the art in motion generation in the context of mobile manipulation. We will identify aspects of motion in mobile manipulation currently not fully addressed and will speculate about how these gaps might be closed.

In the context of mobile manipulation, the generation of robot motion poses a number of challenges, related to those laid out in the introduction to this chapter. Robotic systems able to perform a variety of tasks must possess versatile motion capabilities. This is usually realized through a large number of degrees of freedom, leading to high-dimensional motion generation problems. The humanoid robot Justin, for example, possesses 58 DOFs; the latest version of the Honda ASIMO has 57 DOFs. In addition to the high dimensionality of the associated configuration spaces, uncertainty represents a major challenges when generating robot motion. For applications in mobile manipulation, one simply cannot assume that a precise and complete world model is available at all times. The generation of motion therefore must consider the uncertainties present in the robot's world model. Furthermore, it is unrealistic to assume that the entire world will be perceivable all at once. This implies that the robot's world model will always be partial and potentially wrong in important ways. All this is compounded by the fact that the robots sensors and actuators themselves are prone to uncertainty. It is therefore apt to state that motion generation in mobile manipulation is subject to two of the problems we identified in the introduction: high dimensionality and uncertainty.

In this section, we will examine major branches of research in motion generation and analyze the degree to which they are poised to address the challenges of motion generation in manipulation. But before that, of course, we must discuss what exactly these challenges are.

40.3.1 Problem Description

What are the requirements for robot motion in the context of mobile manipulation? Just as in the classical motion planning problem, a robot must be able to move from one place to another. This requires information about the spatial extent of objects and the resulting global connectivity of world. Algorithms that address this most basic requirement of robot motion are discussed extensively in Chap. 7.

In addition to global, goal-oriented motion, a mobile manipulator must also be able to maintain task constraints during motion: holding a glass of water upright, pointing a camera in an inspection task, or coordinating motion with another robot during cooperative manipulation. At the same time, it must be able to react quickly to unforeseen changes in the environment; global motion planning might not be fast enough and therefore has to be complemented by reactive obstacle avoidance.

While performing this task-consistent, global, and reactive motion, the safe and efficient movements of a mobile manipulator may also depend on kinematic, dynamic, or postural constraints. Postural constraints may be used to perform additional tasks or simply to move in the most energy economical posture to extend battery life.

Finally, all of the motion requirements (global, task-consistent, reactive, joint-limit avoiding, posture-optimizing) must be achieved and maintained in the presence of various sources of uncertainty. In mobile manipulation, there will always be uncertainty about the world model, which might be incomplete or partially

Avidbots Ex1016
Page 1049 of 2154

wrong; changes to the world model may happen without knowing the robot. And, of course, there is uncertainty in regards to sensing, affecting the robot's knowledge of its own state or the ability to predict the outcome of its actions.

As we will see, most existing areas of research in motion generation today cannot address all the aforementioned requirements at the same time. Furthermore, these motion requirements also necessitate sensing capabilities. It is therefore difficult to treat motion generation for mobile manipulation independently from sensing and perception.

40.3.2 Evaluation of the State of the Art

Sampling-based motion planning is the de facto standard for motion planning in robotics; the state of the art in that field is described in detail in appendix of Chap. 7. Here, we want to show some work that extends standard methods, such as probabilistic roadmaps (PRMs) and rapidly-exploring random trees (RRTs), to address some of the motion requirements of mobile manipulation.

Motion Planning

Motion planners have been devised to account for the changes in the configuration space when robots manipulate objects by picking and placing them, possibly handing them off to another robot in order to fulfill a motion task [40.83]. This enables complex sequences of repeated reach and grasps in very tight environments to achieve a planning objective.

Other planners enable the combination of global motion with task constraints [40.84, 85]. The resulting planners generate globally goal-directed motion while maintaining task constraints at the robot's end-effector. This enables collision-free motion while maintaining a fixed orientation of the end-effector, for example. This is realized by identifying lower dimensional, task-consistent manifolds in configuration space and confining possible motions to those manifolds. This is usually achieved through iterative optimization of sampled configurations, adding the computational complexity of global motion planning.

Planners are also able to devise global motion that is consistent with kinodynamic constraints and with actuation limits of robotic platforms. Kinodynamic planning addresses, for example, constraints imposed by nonholonomicity or by the dynamics of the robot [40.86]. Actuation limits become important when a robot is unable to lift a heavy load with its arm fully extended. In such a situation, a planner can generate a motion to slide the object closer to a region of the workspace where the mechanical advantage of the robot kinematics enables the desired motion objective [40.87].

Global motion planning is computationally expensive. Therefore, it generally cannot operate at reactive rates. As a result, they are unable to satisfy the motion requirement of reactive obstacle avoidance. Of course, ideally one would determine a global motion plan at high frequencies; then planners would be able to satisfy global and reactive motion requirements. To overcome the provable computational complexity of planning, one has to resort to approximation algorithms. In other words, one has to trade completeness for computational efficiency. While there are probably many ways of accomplishing this, which may be the subject of future research, one possibility presented in the literature is the balancing of exploration and exploitation [40.88]. By replacing the exploration of motion planners with exploitation on those areas of configuration space where helpful information is available, it is possible to speed up planning by more than one order of magnitude.

Common to all of the motion planning approaches in this section is that they ignore uncertainty. They assume a perfect geometric world model and perfect execution capabilities. These assumptions generally do not hold in mobile manipulation. Hence, it will be difficult to extend traditional motion planning approaches so as to address the entire spectrum of motion requirements described above.

Further information regarding motion generation for mobile manipulation, in particular in the context of humanoid robots, has been collected in a special volume [40.89].

Feedback Motion Planning

Feedback motion planning [40.86, 90] addresses a particular type of uncertainty: actuation uncertainty. With this type of uncertainty, the robot might end up in a different state than it intended. An appropriate plan must therefore contain the correct action to take in every possible configuration the robot might end up in. Such a plan is called a navigation function. Given such a navigation function, the robot can perform Markovian gradient descent until the goal is reached.

Feedback motion planning also assumes a perfect knowledge of the world. The ability to address actuation uncertainty comes at a significant computational cost. Instead of having to compute a single path, an implicit representation of *all* paths has to be determined.

Trajectory Modification

There is an appealing complementarity between motion planning and control methods. While the former are good at addressing global motion constraints, the latter excel at reactive behavior, including obstacle avoid-

ance, force control, maintaining task constraints, etc. If the advantages of these two approaches could be combined, one would have made significant progress toward motion generation for mobile manipulation.

The simplest way of achieving this objective is through consecutive invocation: first determine a path consistent with global motion requirements, then apply control methods to the entire path to continuously optimize in response to changes in the environment or task constraints. This latter part, applying control to an entire path or trajectory as opposed to a single robot configuration, is called incremental trajectory modification.

All methods in this category perform optimization of a function, encoding various desirable properties of the trajectory, through some kind of gradient descent. The earliest method, called elastic bands [40.91], applies repulsive forces from obstacles and attractive internal forces to a discretization of the configuration space trajectory, leading to a behavior reminiscent of elastic material. This method was extended in the elastic strip framework [40.92] by applying these forces directly in the workspace to a discretization of the volume swept by the robot along its configuration space trajectory. Approaching obstacles deform the trajectory; when obstacles recede, the trajectory is shortened by the internal forces.

In contrast to the local reaction forces computed by the elastic band and elastic strip approaches, other approaches optimize trajectories globally in accordance with various criteria of optimality. This optimization problem can be cast as approximate probabilistic inference [40.93], then using message passing algorithms to find the maximum likelihood trajectory for a given objective. Alternatively, the optimization problem can be solved using the Hamiltonian Monte Carlo algorithm [40.94], where covariant functional gradient descent techniques perform local optimization. A stochastic variation of this method seems to be able to deal better with local minima [40.95].

Even though trajectory modification methods are often times labeled as motion planning methods, they do not perform global planning. They have been shown to work well in many realistic scenarios but can fail when gradient information is insufficient to find a solution. As a result, trajectory modification can serve as a powerful component for motion generation, but they cannot address all requirements of motion generation in mobile manipulation.

Integrating Planning and Control

Traditionally, global motion planners produce paths or trajectories that then get passed to the robot for execution. At execution time, these trajectories can be modified, for example, by the methods described in the previous section. It is striking, however, that a planner determines an entire trajectory from start to goal, knowing well that in dynamic environments most of the trajectory will be invalidated within seconds due to the motion of obstacles.

To fully leverage the complementarity of global planning and control mentioned in the previous section, it might be necessary to shift the boundary between these two fields. This would imply that, instead of computing a detailed trajectory, global motion planning would only provide as much information as is required to avoid local minima during control. This leads to the idea of sequencing local controllers, which can be visualized as funnels, in such a way that a global motion task can be achieved [40.96]. Referring to Fig. 40.5, each controller can be viewed as a local plan for region of the configuration space (in the image, the configuration space is the imagined plane above which the funnels are arranged). Each funnel has an exit (the converged state of the controller) that leads into the opening of the next funnel. Arrangements of these funnels represent robust motion plans as each funnel (controller) inherently rejects uncertainty.

Such a shift of the boundary between planning and control is realized in the Elastic Roadmap approach [40.97]. It combines global planning with trajectory modification to yield an approach to motion generation capable of generating global, task-consistent, reactive, and constraint-respecting motion. It does so, however, by deliberately trading completeness for computational efficiency. While this approach works well in practices, no performance guarantees can be made. Also, this approach is not able to address uncertainty; it still assumes that the world and its changes are perfectly known.

Reasoning About Uncertainty

Partially observable Markov decision processes (POMDP) are a general framework to formalize

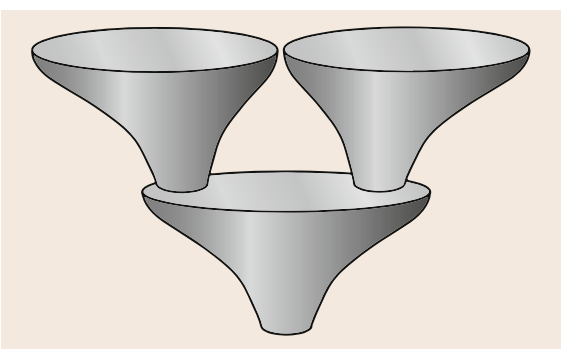


Fig. 40.5 Representing a motion plan as a sequence of funnels

problems involving actions that perform uncertain transition (uncertain outcome) between uncertain (not fully known) states (Chap. 15). POMDPs are so general that they can easily characterize the type of motion planning problem occurring in mobile manipulation. A downside of their generality and expressive power is the computational complexity of finding an exact solution, which is often intractable. To overcome this problem, one must resort to approximations.

Point-based POMDP solvers [40.98] provide such an effective approximation for the case that there is an uncertainty about the robot's state. The key idea of point-based solvers is to sample only parts of the solution space, avoiding regions unlikely to contain a solution [40.99]. Effectively, these methods exploit additional knowledge about the specific problem to guide the search for a solution.

In the context of motion generation, the roadmaps generated by traditional sampling-based motion planners can provide such knowledge for finding solutions to a specific motion planning problem [40.100, 101] or to any motion planning query in a given environment [40.4].

Considering uncertainty not only for the robot's state but also about the world model makes the problem even more difficult. Here also, it becomes necessary to approximate the state space of the world, leading to discrete approximations of the corresponding state space [40.102, 103].

While all the methods mentioned so far successfully address uncertainty, their computational complexity prevents them from being reactive. This means that all initial assumptions about the world, including assumptions about uncertainty distributions (encoded in the POMDP), must be correct and remain correct throughout the entire motion. In the context of mobile manipulation, it is difficult to think that even the initial assumptions could be correct.

If one does not want to rely on fixed assumptions about uncertainty, for example in the state transition function, one has to rely on sensing. For example, a robot might be able to reduce the uncertainty about its own state sufficiently well to be able to avoid reasoning about this type of uncertainty. But this in itself is a very strong assumption. By making such strong assumptions, it is possible to reduce the complexity of addressing uncertainty in a meaningful way, leading to a polynomial time to address uncertainty in the context of mobile manipulation [40.104]. While this approach achieves reactivity, task consistency, and is able to address other constraints, it is incomplete and no performance guarantees exist.

40.3.3 Toward Motion Generation for Mobile Manipulation

The literature contains a number of powerful technical tools and conceptual approaches that have proven effective in the context of motion generation: sampling-based motion planners, planning for feedback, optimization for incremental trajectory modification, sequencing of controllers, shifting the boundary between planning and control, and approximative solutions to POMDP-based formalization of motion planning problems. The use of assumptions to improve the computational efficiency is inherent to all of them – no surprise, given that even simple motion planning problems are already PSPACE-complete.

For each of the aforementioned motion requirements in mobile manipulation, when considered in isolation, an adequate solution exists. But no method or approach addresses all of them in coherent fashion with characterizable performance. Nevertheless, in practice, motion generation capabilities exist for many interesting applications in mobile manipulation.

With respect to the goals laid out in the introduction to this chapter, this discussion has exposed, however, that all approaches still depend on detailed prior knowledge about the environment. Most of this prior knowledge is difficult to obtain in the context of mobile manipulation. For example, accurate geometric models of the environment are unlikely to be obtainable at a global scale. And even if they were attainable, they will likely be outdated very quickly and would require constant updating. It might be even more difficult to obtain adequate characterizations of uncertainty for arbitrary motion problems.

In response to these difficulties, it is highly likely that the close integration of real-world sensing into motion generation will become a more active area of research. This sensing must be used to alleviate the needs for prior world models. This development might parallel the *shifting the boundary* between planning and control, only this time between planning and sensing. Instead of performing planning on very detailed and all encompassing world models, it might be more practical to plan in a coarse and approximate world world, leaving it to perception to fill in the details as they become relevant.

Avidbots Ex1016
Page 1052 of 2154

40.4 Learning

We mentioned high dimensionality, uncertainty, and variability as core challenges for robotic systems that leave prestructured lab environments. Learning is the core approach to cope with the variability of tasks a robot may need to perform in unstructured environments.

The topic of learning is orthogonal to all of the other topics discussed in this section: Learning methods are successfully applied to grasping and manipulation, mobility, control, motion generation, and perception. We will highlight representative approaches in these areas below, but first we want to discuss in more general terms the relation between the areas of machine learning and robotics.

A core question is whether existing machine learning methods can without need for modification be applied in robotic domains. Or whether genuinely novel issues are raised in the context of robot learning that necessarily go beyond core machine learning research. This is fundamentally related to the question *Is there a generic prior*?

In the Bayesian view, learning means computing the posterior

$$P(M|D) = \frac{P(D|M)P(M)}{P(D)} \,. \tag{40.1}$$

A similar discussion could refer to the choice of hypothesis space and regularization. But for simplicity we take the Bayesian view as a reference. The machine-learning community has made great progress in two respects: First, it developed a rapidly growing set of general and fundamental modeling tools to formalize learning problems, including standard parametric and nonparametric models, graphical models, infinite (hierarchical Dirichlet process) models, models combining probabilities with first-order logic, random set models, etc.

Second, the community has developed a large set of methods to solve learning problems, i. e., compute the posterior or optimize discriminative models, including a wide range of probabilistic inference methods and leveraging the tight relations to the field of (constrained) optimization.

With this range of modeling formalisms and computational methods available, what problems are left? Interestingly, a core issue remains the concrete specification of the prior $P(M)$ or hypothesis space itself. An example is the current research on deep representations: *Weston* et al. [40.105], for instance, argue that deep learning can concisely be understood as a form of regularization of intermediate representations, including auto-encoders as special case but generalizing it to arbitrary choices of embeddings. Such a regularization is one-to-one with the choice of model prior in the Bayesian formulation of learning methods. The large majority of (non-deep) regression and classification models are linear in some features (we include Hilbert space features here). The regularization is typically generic: L_2 or L_1 on the respective parameters. Therefore, the model prior is primarily determined by the choice of features.

Researchers generally agree that a core challenge of machine learning is in the specification of the prior $P(M)$ – respectively the choice of features [40.106, 107]. However, is the question about the *right* prior a genuine machine learning question? In other terms: Does there exist a *generic* prior that might perform well in any domain, or is a good choice of prior inherently domain specific, requiring expert knowledge?

Related to this question is the idea of *learning the prior* – or learning the right features. Of course, in the strict sense, a prior must not be data based. When discussing the learning of features, people usually presume a hyperprior that implies a class of potential features. However, it is questionable whether there exist generic hyperpriors over feature classes that would allow systems to learn arbitrary features suited in any domain. Some aspects of priors seem generic and are discussed extensively within machine learning: L_1 or L_2 regularization to enforce sparsity and penalize complexity; metric or topological embedding to enforce the preservation of information in internal (deep) representations. However, the structural complexity of suitable features in real-world domains seems – in the eyes of a roboticist – beyond such generic properties.

In this light, it is not surprising that the success of learning methods within robotics often relies on an insightful way to model the problem using machine learning formalisms combined with a good choice of features. Both depends on the roboticist's expert knowledge and his understanding of appropriate priors in real world domains, contrasting the idea that generic machine learning and priors could be applied for learning in robotic manipulation.

40.4.1 Problem Description

Perhaps a reason for the great success of machine learning is that its problems are – in constrast to robotics – rather straightforward to define rigorously. In most cases a learning problem is defined in terms of minimizing a loss or risk (often approximated using cross

validation), computing a posterior, or maximizing an expected return. Feature and hyperparameter selection are subsumed in these objectives.

To make this more concrete, we choose to detail one specific machine learning approach: conditional random fields (CRF) [40.108]. CRFs have become a default starting point for addressing many learning problems, especially in robotics. And CRFs are a very general way of modeling problems, as we discuss briefly in the following, and should therefore be a standard tool for roboticists.

A CRF could also be named a *conditional graphical model* or *conditional factor graph*. It describes a structured probability distribution over labels y where the factors that define this distribution additionally depend on some input x. The following summarizes the essentials of CRFs:

- We consider output variables $y = (y_1, \dots, y_l)$ and input variables x. A *CRF* is defined by a joint function over x and y of the form

$$f(x, y) = \sum_{j=1}^{k} \phi_j(x, y_{\partial j}) \beta_j = \phi(x, y)^{\mathrm{T}} \beta \,, \tag{40.2}$$

  which describes the predicted output as

$$y^*(x) = \operatorname{argmax}_y f(x, y) \,. \tag{40.3}$$

  Each feature $\phi_j(x, y_{\partial j})$ depends on a subset $y_{\partial j}$ of output variables, $\partial j \subseteq \{1, \dots, l\}$. The function f is also called (neg-) *energy* or *discriminative function*. CRFs can be trained based on a probabilistic interpretation or with a hinge loss.
- In the *probabilistic interpretation* [40.108], CRFs define a conditional probability distribution over y,

$$p(y|x) = \frac{\mathrm{e}^{f(x,y)}}{\sum_{y'} \mathrm{e}^{f(x,y')}} = \mathrm{e}^{f(x,y) - Z(x,\beta)} \tag{40.4}$$

$$= \mathrm{e}^{-Z(x,\beta)} \prod_{j=1}^{k} \mathrm{e}^{\phi_j(x, y_{\partial j}) \beta_j} \,, \tag{40.5}$$

  which is a Boltzmann distribution of f with log-partition function

$$Z(x, \beta) = \log \sum_{y'} \mathrm{e}^{f(x,y')}$$

  ensuring the conditional normalization. This is trained by minimizing

$$\beta^* = \operatorname{argmin}_\beta L(D, \beta) + \lambda \|\beta\| \,, \tag{40.6}$$

  where the second term is an L_2 (*Ridge*) or L_1 (*Lasso*) regularization and the first the neg-log-likelihood on data $D = \{(x_i, y_i)\}_{i=1}^N$. The optimum is found using a Newton method, using the following gradient and Hessian terms:

$$L(D, \beta) = -\sum_i \log p(y_i|x_i)$$

$$= -\sum_i [\phi(x, y)^{\mathrm{T}} \beta - Z(x_i, \beta)] \,, \tag{40.7}$$

$$\nabla_\beta Z(x, \beta) = \sum_y p(y|x)\, \phi(x, y) \,, \tag{40.8}$$

$$\nabla_\beta^2 Z(x, \beta) = \sum_y p(y|x)\, \phi(x, y)\, \phi(x, y)^{\mathrm{T}}$$

$$- \nabla_\beta Z\, [\nabla_\beta Z]^{\mathrm{T}} \,. \tag{40.9}$$

  Computing $p(y|x_i)$ (or the factor marginals $p(y_{\partial j}|x_i)$) for all x_i is analogous to the E-step of expectation maximization; the parameter update of β in each Newton iteration is analogous to the M-step. For arbitrary structures, the Hessian term $\sum_y p(y|x)\phi(x, y)\phi(x, y)^{\mathrm{T}}$ may be infeasible to compute exactly. Still, Newton methods with approximate Hessian may be significantly faster than gradient-based methods.
- In the *discriminative interpretation* [40.109], the CRF is trained by minimizing a *hinge loss*

$$\min_{\beta, \xi} \; \beta^2 + C \sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \;\; \forall_{y \neq y_i} : f(x_i, y_i) - f(x_i, y) \geq 1 - \xi_i, \;\; \xi_i \geq 1 \,, \tag{40.10}$$

  using either perceptron or linear programming methods similar to how SVMs use them [40.109]. This approach is also called *structured output SVM*.
- *Kernelization* works for both types of training. Further, in the probabilistic interpretation, the ridge regularization can be replaced by a Gaussian prior $P(\beta) = \mathcal{N}(\beta \mid 0, \sigma^2/\lambda)$ and a fully *Bayesian* predictive posterior $P(y|x, D)$ can be computed (just as for Bayesian ridge regression and Gaussian processes).
- CRFs are a very general class of learning tool. We list *special cases*, where the options concern the special case input and output spaces, *prob*abilistic or *hinge* loss training, *kernel*ization, *bayes*ianization, and of course the choice of features (explained below):
  SVM = CRF(kernel, hinge, $y \in \{0, 1\}, \phi_C$)
  logistic regression = CRF(prob, $y \in \{0, 1\}, \phi_C$)

Gaussian process (GP) = CRF(prob, bayes, kernel, $y \in \mathbb{R}$, ϕ_R).
GP classification = CRF(prob, bayes, kernel, $y \in \{0,1\}$, ϕ_C)
factor graph (MRF) = CRF(prob, $x = \emptyset$)
hidden Markov model (HMM) = CRF(prob, $y = (y_1, \ldots, y_T)$, $\{\phi_t\}$)
ridge regression = CRF(prob, $y \in \mathbb{R}$, ϕ_R)
The regression case is somewhat special, see below.

- We give three *feature examples* for the cases of multiclass classification, Markov chains (like HMMs), and the somewhat special case of regression.
For a three-class classification, one chooses

$$\phi_C(y,x) = \begin{pmatrix} [y=1]\phi(x) \\ [y=2]\phi(x) \\ [y=3]\phi(x) \end{pmatrix} \in \mathbb{R}^{3k}$$

where $y \in \{1,2,3\}$, $[expr.] \in \{0,1\}$ is the indicator function and $\phi(x) \in \mathbb{R}^k$ an arbitrary (e.g., polynomial) input feature vector. The effect of the indicators is that for each class, a different subset of β-elements define the function f.
For a Markov chain $y = (y_1, \ldots, y_T)$ with binary y_t, one chooses

$$\phi_t(y_{t+1}, y_t, x) = \begin{pmatrix} [y_t = 0 \wedge y_{t+1} = 0] \\ [y_t = 0 \wedge y_{t+1} = 1] \\ [y_t = 1 \wedge y_{t+1} = 0] \\ [y_t = 1 \wedge y_{t+1} = 1] \\ [y_t = 1]\phi_t(x) \end{pmatrix} \in \mathbb{R}^{4+k},$$

where $\beta_{1:4}$ determines the transition probabilities, and $\beta_{5:5+k-1}$ plays the same role as in the logistic regression of y_t on input features $\phi_t(x) \in \mathbb{R}^k$. If $x = (x_1, \ldots, x_T)$ is also a time series and if we choose input features $\phi_t(x) = \phi_t(x_t)$ to depend only on x_t, we are in the case of HMMs. The great strengths of Markov chain-CRFs over HMMs is that $\phi_t(x)$ may depend arbitrarily on the whole of x. In the (unusual) regression case, we may choose

$$\phi_R(y,x) = \begin{pmatrix} -\frac{1}{2}y^2/\sigma^2 \\ \phi(x)y/\sigma^2 \end{pmatrix} \in \mathbb{R}^{1+k},$$

which leads to $y^*(x) = \phi(x)^{\mathrm{T}}\beta$ and $p(y|x) = \mathcal{N}(y \mid y^*(x), \sigma^2)$ with a partition function independent of x and β. Therefore, a single Newton step leads to the well-known optimal parameters of ridge regression. GPs are the bayesian kernel variant of this.

An extreme special case is logistic regression, where the graphical model only concerns a single binary output variable y, the distribution over which depends on the input x. On the other side, also any kind of standard (non-conditional) graphical model over y is an extreme special case of a CRF that lacks the dependence on any external x. Conditional Markov chains (e.g., in linguistics), Markov random fields (in computer vision), and many other models can be viewed as instances of CRFs.

Given this concrete framework of CRFs, the problem – from the robotics perspective – essentially becomes to propose an appropriate graphical model structure over y, and to choose appropriate features $\phi(x,y)$. Both choices together correspond to the choice of hypothesis space or prior as discussed above.

40.4.2 Evaluation of the State of the Art

Perhaps the best developed area for the application of learning methods in robotics is perception.

Learning for Perception and Scene Understanding

Saxena et al. developed a series of methods that very successfully employ the general framework of CRFs in various perception tasks. In [40.110] they train CRFs to predict the depth in two-dimensional (2-D) views; [40.111] learns CRFs to classify a *grasp*-affordance in the 2-D view; [40.112] learns CRFs (or SVMs) to classify the *place*-affordance of locations; [40.113] learns CRFs to label and thereby anticipate the actions and movements of demonstrators. These works are exemplary for the power of current machine-learning methods for perception – but also in that their success relies on the creative choice of problem formalization and features. As an example, we would like to mention the unique features chosen to be indicative of concavity and place stability to predict cup-like place locations in [40.111]. Could such types of features ever be discovered autonomously by more advanced machine learning methods?

Also, for 3-D point cloud data the definition of clever features turned out to be essential for basic registration and classification problems [40.114].

With respect to our discussion of learning features, in standard image classification tasks there has recently been great success in using generic (sparse coding) priors to learn features that significantly outperform hand-coded features (like robust features (SURF) and histograms of oriented features (HOG) [40.115]. Such learned features correspond to local 2-D or 3-D appearance patterns (which define the codebook); the histogram of such patterns turns out to be an excellent basis for image classification. It seems promising that

typical 2-D CRF approaches to classify affordances – as those mentioned above – can be extended to also use learned local features. However, there may be limitations to such generically learned features. For instance, naively they do not respect any natural invariances (such as affine or illumination invariance; this can be fixed by artificially incorporating such invariances in the dataset). Using merely sparse coding priors, they are unlikely to reflect stronger priors (e.g., Gestalt laws, or kinematic understanding of rigid body scenes) of natural physical environments. For instance, the concavity feature for placement prediction seems in strong contrast to such learned features.

As a conclusion with respect to perception, the existing machine learning machinery, including novel feature learning methods have shown great success. For specific mobile manipulation tasks (like affordance prediction) this often relied on well chosen problem formalization. As is the case also in the following topics, a core question for future research are novel ideas for feature classes and priors, that are generic and support the learning of features for manipulation tasks, but incorporate stronger physical world priors than existing approaches.

Learning for Grasping

The ability to grasp is essential to manipulation and approaches to learning to grasp therefore deserves special attention. Learning to grasp has been addressed from at least two perspectives.

First, on the perceptual side, several authors have proposed methods to directly classify 2-D or 3-D appearance patterns that predict a grasp affordance [40.111]. On the positive side, this implies a very close perception–action cycle; the percept is directly translated to a potential pregrasp location without need to estimate a 3-D shape of the object or even a notion of objects and shapes. Such approaches provide a sort of heat map of potential affordances, e.g., for exploration or to reduce the number of options for manipulation. But they lack the generality of grasp approaches that take into account the goal of applying a certain force or motion to an object – but classically require a shape estimate.

Second, concerning the grasp motion itself, a basic approach is to employ policy search methods to improve a parametric dynamic movement primitive (DMP) model of the grasp motion [40.116].

Learning for Motion Generation

There exists extensive work on learning methods for control, e.g., in the form of system identification, (model-based and model-free) reinforcement learning and learning from demonstration. It is beyond the scope of this chapter to cover this. Instead, here we focus on learning in the context of path finding or trajectory planning for manipulation.

A core issue of motion generation for manipulation is the question of the right task space. For instance, the work by *Cakmak* and *Thomaz* on learning from human interaction [40.117] suggests that the question of what is the actual task space should be in the center of communication. In a certain formalization, the choice of the task space of a motion is perfectly analogous to the choice of features in machine learning: The mapping from a geometric problem setting to an appropriate motion can be formalized as a CRF [40.118], where the optimal motion maximizes a discriminative function (or neg-cost function). Typically such a cost function is a sum of squares of features, where each feature captures some error in some nonlinear task space. The problem of finding appropriate task spaces – the right features of motion – is therefore analogous to the problem of finding appropriate features in machine learning. And as for generic machine learning, the core questions are: what is a general class of potential features (of potential task spaces), and what is a possible prior over this class.

Jetchev and *Toussaint*'s task space retrieval (TRIC) approach [40.119] considers a combinatorial set of potential task dimensions that refer to various absolute and relative coordinates of objects and uses a Lasso prior to select those features that best explain a set of demonstrated trajectories. While this successfully extracts the ground-truth task spaces w.r.t. which the given trajectories have been optimized, the approach still seems limited by the crude set of allowed features. Relative coordinates (w.r.t. object centers) cannot express more meaningful and better generalizing concepts, e.g., fitting one piece into another. Further, features of manipulation motions might refer to the temporary interaction with contacts or other environmental constraints [40.120, 121]. It should be the subject of future research how such aspects could span a well-formalized set of features that leads to an appropriate prior over manipulation motions and respective optimization methods.

The above discussion views motion as being modeled by a cost (or discriminative) function. This is close to the view of motion as being modeled by an attractor (Lyapunov or potential) function; a cost function implies such a potential in the form of the cost-to-go function, and shifting the modeling directly to the cost-to-go function avoids the optimization step. The core issue of finding the right features for modeling the potential remains unchanged. TRIC [40.119] in fact models on the level of the cost-to-go instead of the cost function.

However, these views on modeling motion seem orthogonal to sample-based path finding approaches, as they traditionally operate only in configuration space and employ a discrete notion of feasible/infeasible and goal region.

Learning Sequential Manipulation from Demonstration

It is fairly recent that learning has also been applied on the level of higher level sequential manipulation. *Niekum* et al. [40.122] present a complete system to analyze a demonstrated sequential manipulation and *replay* it in a robust way on a robot, including its ability to search for alternative steps if some manipulation failed. Again, the choice of representation is the heart of this method: in [40.122] skill trees are used to represent the learned policy. *Alexandrova* et al. [40.123] extend such approaches by proposing an explicit user interface that allows the demonstrator to correct the robot's interpretation of the task and e.g., explicitly communicate the relevant task spaces.

Learning sequential manipulation is structurally one of the most interesting research challenges as it eventually requires choices of representations and priors on all levels, from perception and motion generation to the higher level representation of action and cooperation policies. The abovementioned work does first steps in the context of learning from demonstration. But much research needs to be done to enable robots to autonomously acquire skills that would allow them to control and manipulate the state of their environment as dexterously as we humans do.

40.4.3 Toward Learning in Mobile Manipulation

We above raised the question whether existing machine learning methods can without need for modification be applied in robotic domains, or whether genuinely novel issues are raised in the context of robot learning that necessarily go beyond core machine learning research. The above examples give two answers to this: First, standard machine learning tools like CRFs are widely and very successfully applied within the area mobile manipulation, employing off-the-shelf algorithms for training. Second, however, the above examples all show that the actual crux is in the modeling of the learning problem itself, in particular in the choice of features, the used regularization and the chosen structure of the output. These are issues that require substantial domain expertise. In some cases, modern approaches to feature learning or feature selection are indeed successful – particularly impressive are novel sparse coding principles to train perceptual features [40.115]. However, in many cases successful learning for mobile manipulation requires that a very specific feature space has been chosen based on the roboticist's insight (e.g., the potential operational spaces in TRIC, or the shape concavity feature to predict placement).

If we had a more fundamental theory of real-world mobile manipulation problems, we could hope that such a theory would also inform us on what are promising and general feature spaces and priors for learning in manipulation.

40.5 Perception

The need for perception is one of the main culprits in making problems in manipulation high dimensional. Technological as well as biological agents possess many sensors to perceive their environment. Humans, for example, possess on the order of 300 million nerve endings that produce a rich, continuous stream of information about the world, which is preprocessed and ultimately routed to the brain. The same is true in mobile manipulation: it is not uncommon for robots to be equipped with several cameras, each having millions of pixels, and each contributing to the high dimensionality of the input space.

To illustrate this point: A simple, black and white (not gray scale!) camera with only one thousand pixels can sense many, many more distinct images (namely, $2^{1000} \approx 10^{310}$) than there are atoms in the observable universe (about 10^{80}) or than nanoseconds have elapsed since the big bang (about 4.354×10^{26}). Even if every atom in the universe would generate one such image every nanosecond, all possible images would not have been generated by the time the universe dies its postulated heat death in 10^{100} years. Biological agents can cope with this high dimensionality, of course, because there is substantial structure in the world, i. e., many of the possible images simply do not occur in nature or some of the variability in the data is not relevant for the task at hand and can be ignored.

In this section, we will survey different successful approaches to perception for mobile manipulation that leverage this structure in perceptual data. We will mostly focus on visual perception, reflecting the emphasis of ongoing research activities. The foundations of perception in robotics are presented in great detail in Part C of this Handbook. Chapter 42 also contains relevant approaches to perception.

40.5.1 Problem Description

Perception is the process of interpreting sensory data to enable useful actions. In the context of robotics, this may be – given the state of the art – the most suitable, albeit very broad definition of the perception problem. It differs appropriately from the definition most commonly used in psychology in that the goal of perception is not solely to gain an understanding of the world, but also to use such an understanding to enable task-directed actions. This modification in fact can be seen as a simplification of the original perception problem, as only those aspects of the environment must be perceived that are relevant to a particular action. And, in fact, there is evidence from the human perceptual system that such a limitation does indeed take place, and sometimes leads to odd perceptual deficiencies [40.124].

In the context of robotics, the earliest perception work followed the psychologists' definition of perception. It focused on the use of visual perception to derive an exact 3-D model of the thing being perceived from its two-dimensional optical projection onto a camera sensor [40.125]. However, this problem is mathematically ill-posed (three-dimensional shape information cannot be reconstructed unambiguously from its two-dimensional projection) and, as we argued above, also high dimensional. To enable robust and competent perception in the context of mobile manipulation, it therefore seems sensible to attempt to reduce the complexity of perception, lending further support to our definition of perception above.

40.5.2 Evaluation of the State of the Art

Much of the work we will discuss in the remainder of this section follows the robotics-specific view of the perception problem given above. However, we will start out by surveying impressive advances made toward the objective of reconstructing three-dimensional, geometric models of the environment. Only the future can tell us if these methods are well suited to supporting the kind of complex and general manipulation tasks encountered in mobile manipulation. In the subsequent sections, we will then cover alternative approaches.

3-D Mapping and Localization

Recently, there has been remarkable progress in dense 3-D mapping and visual SLAM. What originated with DTAM [40.126] lead to current systems like LSD-SLAM [40.127] which efficiently and rather robustly estimates a 3-D environment model in real-time, without GPU, from monocular video.

Efficient Dense and Multiresolution Mapping

Recent multiscale surfel models [40.128, 129] using RGB-D sensors are an alternative to directly pixel-based reconstruction methods and provide a similarly dense 3-D model. In [40.130] and [40.131], a 3-D representation, Octree, is proposed having multiple resolutions based on a probabilistic approach. This 3-D multiresolution method can provide flexible occupancy mapping depending on the demand of the high or low resolutions. *Saarinen* et al. [40.132] proposed normal distributions transform occupancy map (NDT-OM), which has the compactness from NDT maps and the robustness from occupancy maps. This algorithm supports for multiresolution maps and it efficiently enables mapping in dynamic environments.

These developments imply that, leaving system integration problems aside, we may today assume to have access to precise 3-D mesh models of the environment. In the concrete mobile manipulation context 3-D information is needed with high resolution only when the robot is near the object or the environment. Therefore, practical techniques for multiresolution map are developed based on the distance from the robot. *Droeschel* et al. [40.133, 134] show 3-D multiresolution maps where sparse resolution is used for the environment far away and high resolution is used for the object to be interacted soon.

Mapping and Tracking with Moving Objects

Most methods for 3-D mesh estimation assume a static environment (but moving sensor). The segmentation into objects and dealing with moving objects – or rather exploiting the movement of objects – is left aside. However, fortunately recent approaches adopt a more integrated view on perception by jointly addressing the problems of dense 3-D reconstruction, moving object segmentation and tracking, and kinematic joint perception [40.135, 136], similarly as has previously been done on the basis of key points. We think that such integrated perception systems, fusing visual evidences and priors on multiple levels of representations (from low-level dense pixels to high-level rigid body hypotheses) should be the core effort of future research in this area.

Other approaches deal with general dynamic environments by efficiently updating the map continuously. Especially, in the real world problem, there are always moving obstacles and the environment itself can change over time. A incremental update algorithm is proposed in [40.137] for grid-based spatial representations for dynamic environment due to moving obstacles. Distance maps, Voronoi diagrams, and configuration-space collision maps are updated with the algorithm for the changes in the environment. In [40.138], velocity

occupancy space is proposed by adding the velocity obstacle concept into occupancy grid. This is to enable the robot to efficiently navigate through uncertain moving obstacles.

Indoor/Outdoor

Finally, the usage of a map can be different for indoor or outdoor applications. The outdoor map is mainly used for navigation whereas the indoor map is for both navigation and manipulation. The map for outdoor or field applications needs to cover wider range of area for navigation and the mobile platforms do not require high resolution maps compared to indoor applications. Indoor applications, however, typically require higher resolution even for navigation because there are more obstacles and less free space. Because of this space restriction, the navigation route needs to be planned carefully considering the obstacles. Manipulation tasks are more involved in the indoor applications such as cleaning, connecting hoses, and locking a valve, which tasks must require much higher resolution than typical navigation tasks.

For such different applications, the maps for indoor and outdoor can be quite different. Thus, in [40.139] pictures are used for the classification of the current environment. Specifically, Hue color component and color temperature values from each image are used with k-nearest neighbor (KNN) classifier to classify the environment into indoor or outdoor. *Payne* and *Singh* [40.140] proposed a simple system for the indoor–outdoor classification, which can be used as a real-time system. It is based on the assumption that indoor images contain more straight lines that outdoor environments.

Differently from indoors, the outdoor map often requires terrain information. In [40.141], the system classify if the current environment is indoor or outdoor, then terrain mapping is used for outdoor environment. It also provides seamless integration of mapping between indoor and outdoor environment.

Active and Interactive Perception

The success of the 3-D mapping approaches presented in the previous section is enabled by camera motion. This motion provides a continuous stream of incrementally changing two-dimensional projections of the three-dimensional scene. This enriched perceptual information helps resolve the ambiguities of the projection from 3-D to 2-D, thus overcoming limitations of *static* vision approaches.

Perceptual algorithms that deliberately change the sensor to optimize the perceptual processes are called *active*. Among active perception methods, active vision is probably the most prominent one [40.142–144]. But active haptic perception methods also hold promise as a perceptual approach for mobile manipulation. Active haptic perception has been used for object localization [40.145, 146], contact point localization [40.147], and for estimating the object's shape [40.148].

The concept of active perception can be extended to include interactions with the environment. Now the observer is able to manipulate the environment to reveal additional perceptual information, either by removing obstructions or by creating sensory signals that would otherwise not be observable. The class of perceptual algorithms relying on motion of the sensor as well as the deliberate application of forces to the environment is referred to as *interactive perception*.

Interactive perception eliminates the traditional boundary between perception and manipulation, combining both into a tightly coupled feedback loop. Manipulation now becomes an integral part of perception – in addition to being an objective in itself. At the same time, perception of task-relevant information is now critically enabled by manipulation. This combined perception-and-manipulation process links inseparably the goals of achieving a manipulation task and of acquiring perceptual information. These two goals must be balanced appropriately, as it is not desirable to perceive the world without progress toward the task, and it most often is not possible to achieve the task without perceiving something about the world first. Now, in addition to perception and manipulation, we must also consider aspects of machine learning: how do we balance progress toward the task (exploitation) with the goal of acquiring sufficient information to reach the goal robustly and reliably (exploration)? These and many related questions are current topics of research in this emerging field at the intersection of perception, manipulation, and machine learning.

Using interactive perception (or perceptual manipulation, if one prefers), it is possible to separate piles of objects so that they can be perceived, manipulated, and sorted [40.149–151]. It is also possible to interactively search for objects by removing obstructions [40.152]. Using interactive perception, it becomes possible to reveal sensory information about the articulations of objects in the environment, i. e., about their inherent degrees of freedom [40.153]. This ability is a perceptional prerequisite for mobile manipulation, as the degrees of freedom of an object are generally linked to its function. For example, by pushing on a door handle and observing its motion, it is possible to perceive the revolute joint connecting the handle to the door. Operating this revolute joint is necessary to open the door.

Interactive perception approaches based purely on vision data exist for planar objects [40.154] and three-

dimensional articulated objects [40.155, 156]. Their performance can be improved significantly by relying on RGB-D data [40.157, 158]. Interactive perception methods have been integrated with object databases so as to recognize previously seen objects and the associated kinematic model [40.159]. Online interactive perception enables the integration of interactive perception directly into the manipulation process [40.153]. This permits to detect failures, monitor progress, and identify the successful completion of a manipulation action. It is also possible to categorize novel types of articulation with interactive perception [40.160, 161]. Interactive perception can include additional modalities, such as proprioception and audition, for example, to identify objects [40.162].

One of the challenges remains the selection of appropriate actions for interactive perception. There are many possible ways to interact with the environment but only few of them will reveal relevant information. The selection of promising actions can be learned from experience [40.163] or by entropy-driven exploration [40.164]. The information obtained from such exploration can be represented in relational knowledge about the world [40.165].

The work discussed above has demonstrated that interactive perception is a promising approach to perception *and* manipulation in the context of mobile manipulation. It has resulted in robust perception that relies on few assumptions about the world, and in particular does not rely on the existence of prior geometric models. It also has resulted in robust manipulation, supported by the most task-relevant perceptual feedback. Interactive perception eliminates the boundaries between perception, action, and learning. While this seems like a complication, the close integration of these different aspects of robot behavior lead to novel solutions of long-standing challenges in perception and manipulation.

Affordance Detection

An affordance can be viewed as an opportunity of an agent to perform an action in its environment. The term dates back to 1977, when it was introduced by psychologist *Gibson* [40.166]. Since then the term has been used inconsistently with slight variations of meaning [40.167].

Here, we regard an affordance as an opportunity to instantiate a behavioral capability of the agent. Such an instantiation of a pushing action, for example, selects the objects involved (the robot's hand and a door handle within reach) and determines other free parameters of the behavior (pushing down on one side of the door handle).

It is appealing to focus perception of affordances as, by definition, this will focus perception only on those aspects of the environment that must be known to instantiate and successfully execute a certain behavior.

Research on affordance perception is intimately linked with research on affordance learning [40.168–170]. A modern view perspective is to consider affordance learning for manipulation as a statistical relational learning problem [40.171].

More low-level affordance perception research considers the problem of labeling scene segments by their affordance of certain actions, often using conditional random fields (CRFs) to train the labeled segmentation. This has extensively been addressed in the case of learning to detected grasp affordances [40.172–175].

40.5.3 Toward Robust Perception

Perception is one of the great challenges in mobile manipulation. We recall that the objective of mobile manipulation research is to a) maximize task generality of autonomous robotic systems, while at the same time b) minimizing the dependence on task-specific, hard-coded, or narrowly-relevant information. Given these objectives, it becomes clear that the complex and diverse information required for task generally must be acquired through perception. And perception must also compensate for the envisaged reduction of a priori knowledge available to the agent. Perception plays a central role in realizing mobile manipulation.

In this section, we surveyed some approaches designed to meet the challenges posed by the perception requirements of mobile manipulation. While a multitude of promising approaches exist, today there is no generally accepted framework or philosophy for machine perception. Evidence from human perception may suggest that such a common framework exists. Some results seem to imply that there are some shared working principles between vision and audition [40.3]. Irrespective of this, there is strong evidence that human perception is built to support action. This becomes apparent already in the retina, which is far from being a general sense organ; it contains highly specialized circuitry tailored for task-relevance [40.176]. Maybe this too can be a clue for how to address perception in robotics: Maybe we need much better low-level features?

40.6 Conclusions and Further Reading

While mobile manipulation requires the integration of methodologies from almost all aspects of robotics – we mentioned grasping, control, motion generation, learning and perception – it also suggests alternative approaches to the state of the art in these areas of research. Instead of considering each research area as independent, research in mobile manpulation exploits their interdependence when this simplifies the overall problem. Active and interactive perception, i. e., combining manipulation and 3-D perception, is one example that departs from the goals of the individual subareas: (1) general 3-D reconstruction from passive data and (2) manipulation planning given a 3-D environment model. Instead, these approaches view both as an integrated and potentially simpler problem.

In the present survey we only briefly covered the various aspects and their interdependencies. Which of the approaches will turn out to be the most successful for the integrated overall goal of mobile manipulation is an ongoing debate. An interesting starting point for further reading are the various workshops that focus on the integrated nature of mobile manipulation and robotics as a whole. A list of these workshops is maintained on the Website of the Technical Committee on Mobile Manipulation.

Video-References

VIDEO 650 Learning dexterous grasps that generalize to novel objects by combining hand and contact models
available from http://handbookofrobotics.org/view-chapter/40/videodetails/650

VIDEO 651 Atlas whole-body grasping
available from http://handbookofrobotics.org/view-chapter/40/videodetails/651

VIDEO 652 Handle localization and grasping
available from http://handbookofrobotics.org/view-chapter/40/videodetails/652

VIDEO 653 Catching objects in flight
available from http://handbookofrobotics.org/view-chapter/40/videodetails/653

VIDEO 654 Avian-inspired grasping for quadrotor micro UAVs
available from http://handbookofrobotics.org/view-chapter/40/videodetails/654

VIDEO 655 A compliant underactuated hand for robust manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/655

VIDEO 656 Yale Aerial Manipulator – Dollar Grasp Lab
available from http://handbookofrobotics.org/view-chapter/40/videodetails/656

VIDEO 657 Exploitation of environmental constraints in human and robotic grasping
available from http://handbookofrobotics.org/view-chapter/40/videodetails/657

VIDEO 658 Adaptive synergies for a humanoid robot hand
available from http://handbookofrobotics.org/view-chapter/40/videodetails/658

VIDEO 660 Universal gripper
available from http://handbookofrobotics.org/view-chapter/40/videodetails/660

VIDEO 661 DLR's Agile Justin plays catch with Rollin' Justin
available from http://handbookofrobotics.org/view-chapter/40/videodetails/661

VIDEO 662 Atlas walking and manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/662

VIDEO 664 Dynamic robot manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/664

VIDEO 665 CHOMP trajectory optimization
available from http://handbookofrobotics.org/view-chapter/40/videodetails/665

VIDEO 667 Motor skill learning for robotics
available from http://handbookofrobotics.org/view-chapter/40/videodetails/667

VIDEO 668 Policy learning
available from http://handbookofrobotics.org/view-chapter/40/videodetails/668

VIDEO 669 Autonomous robot skill acquisition
available from http://handbookofrobotics.org/view-chapter/40/videodetails/669

VIDEO 670 State representation learning for robotics
available from http://handbookofrobotics.org/view-chapter/40/videodetails/670

VIDEO 671 Extracting kinematic background knowledge from interactions using task-sensitive relational learning
available from http://handbookofrobotics.org/view-chapter/40/videodetails/671

VIDEO 673 DART: Dense articulated real-time tracking
available from http://handbookofrobotics.org/view-chapter/40/videodetails/673

VIDEO 674 Reaching in clutter with whole-arm tactile sensing
available from http://handbookofrobotics.org/view-chapter/40/videodetails/674
VIDEO 675 Adaptive force/velocity control for opening unknown doors
available from http://handbookofrobotics.org/view-chapter/40/videodetails/675
VIDEO 676 Interactive perception of articulated objects
available from http://handbookofrobotics.org/view-chapter/40/videodetails/676
VIDEO 776 A day in the life of Romeo and Juliet (Mobile Manipulators)
available from http://handbookofrobotics.org/view-chapter/40/videodetails/776
VIDEO 782 Flight stability in aerial redundant manipulator
available from http://handbookofrobotics.org/view-chapter/40/videodetails/782
VIDEO 783 HERMES, A humanoid experimental robot for mobile manipulation and exploration services
available from http://handbookofrobotics.org/view-chapter/40/videodetails/783
VIDEO 784 Task consistent Obstacle avoidance for mobile manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/784
VIDEO 785 Handling of a single object by multiple mobile robots based on caster-like dynamics
available from http://handbookofrobotics.org/view-chapter/40/videodetails/785
VIDEO 786 Rolling Justin – a platform for mobile manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/786
VIDEO 787 Combined mobility and manipulation – operational space control of free-flying space robots
available from http://handbookofrobotics.org/view-chapter/40/videodetails/787
VIDEO 788 Mobile robot helper
available from http://handbookofrobotics.org/view-chapter/40/videodetails/788
VIDEO 789 Free-floating autonomous underwater manipulation: Connector plug/unplug
available from http://handbookofrobotics.org/view-chapter/40/videodetails/789
VIDEO 790 Development of a versatile underwater robot – GTS ROV ALPHA
available from http://handbookofrobotics.org/view-chapter/40/videodetails/790

References

40.1 R.A. Brooks, J. Connell, P. Ning: *Herbert: A Second Generation Mobile Robot*, Technical Report, AI Memo 1016 (MIT, Cambridge 1988)
40.2 O. Khatib, K. Yokoi, O. Brock, K. Chang, A. Casal: Robots in human environments: Basic autonomous capabilities, Int. J. Robotics Res. **18**(7), 684–696 (1999)
40.3 R.J. Adams, P. Sheppard, A. Cheema, M.E. Mercer: Vision vs. hearing: Direct comparison of the human contrast sensitivity and audibility functions, J. Vis. **13**(9), 870 (2013)
40.4 A.-A. Agha-Mohammadi, S. Chakravorty, N.M. Amato: Firm: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements, Int. J. Robotics Res. **33**(2), 268–304 (2013)
40.5 A. Miller, P. Allen: Graspit! a versatile simulator for robotic grasping, IEEE Robotics Autom. Mag. **11**(4), 110–122 (2004)
40.6 A. Rodriguez, M.T. Mason: Grasp invariance, Int. J. Robotics Res. **31**(2), 236–248 (2012)
40.7 S. Hirose, Y. Umetani: The development of soft gripper for the versatile robot hand, Mech. Mach. Theory **13**(3), 351–359 (1978)
40.8 A.M. Dollar, R.D. Howe: The highly adaptive SDM hand: Design and performance evaluation, Int. J. Robotics Res. **29**(5), 585–597 (2010)
40.9 M. Ciocarlie, F.M. Hicks, S. Stanford: Kinetic and dimensional optimization for a tendon-driven gripper, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 217–224
40.10 L.U. Odhner, L.P. Jentoft, M.R. Claffee, N. Corson, Y. Tenzer, R.R. Ma, M. Buehler, R. Kohout, R.D. Howe, A.M. Dollar: A compliant, underactuated hand for robust manipulation, Int. J. Robotics Res. **33**(5), 736–752 (2014)
40.11 M.G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, A. Bicchi: Adaptive synergies for the design and control of the Pisa/IIT SoftHand, Int. J. Robotics Res. **33**(5), 768–782 (2014)
40.12 J. Amend, E. Brown, N. Rodenberg, H. Jaeger, H. Lipson: A positive pressure universal gripper based on the jamming of granular material, IEEE Trans. Robotics **28**(2), 341–350 (2012)
40.13 R. Deimel, O. Brock: A compliant hand based on a novel pneumatic actuator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) Karlsruhe (2013) pp. 472–480
40.14 M. Kazemi, J.-S. Valois, J.A.D. Bagnell, N. Pollard: *Robust object grasping using force compliant motion primitives*, Techn. Rep. CMU-RI-TR-12-04 (Carnegie Mellon University Robotics Institute, Pittsburgh 2012)
40.15 M. Dogar, S. Srinivasa: Push-grasping with dexterous hands: Mechanics and a method, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2123–2130
40.16 L. Chang, G. Zeglin, N. Pollard: Preparatory object rotation as a human-inspired grasping strategy, IEEE-RAS Int. Conf. Humanoids (2008) pp. 527–534
40.17 N. Chavan-Dafle, A. Rodriguez, R. Paolini, B. Tang, S. Srinivasa, M. Erdmann, M.T. Mason, I. Lund-

Avidbots Ex1016
Page 1062 of 2154

berg, H. Staab, T. Fuhlbrigge: Extrinsic dexterity: In-hand manipulation with external forces, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)
40.18 T. Lozano-Pérez, M.T. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robots, Int. J. Robotics Res. **3**(1), 3–24 (1984)
40.19 M.T. Mason: The mechanics of manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1985) pp. 544–548
40.20 M. Kaneko, T. Shirai, T. Tsuji: Scale-dependent grasp, IEEE Trans. Syst. Man Cybern., A: Syst. Hum. **30**(6), 806–816 (2000)
40.21 R. Deimel, C. Eppner, J.L. Ruiz, M. Maertens, O. Brock: Exploitation of environmental constraints in human and robotic grasping, Proc. Int. Symp. Robotics Res. (2013)
40.22 M.R. Cutkosky: On grasp choice, grasp models, and the design of hands for manufacturing tasks, IEEE Trans. Robotics Autom. **5**(3), 269–279 (1989)
40.23 T. Feix, R. Pawlik, H. Schmiedmayer, J. Romero, D. Kragic: A comprehensive grasp taxonomy, Robotics Sci. Syst.: Workshop Underst. Hum. Hand Adv. Robotics Manip. (2009)
40.24 M. Santello, M. Flanders, J. Soechting: Patterns of hand motion during grasping and the influence of sensory guidance, J. Neurosci. **22**(4), 1426–1435 (2002)
40.25 V.N. Christopoulos, P.R. Schrater: Grasping objects with environmentally induced position uncertainty, PLoS Comput. Biol. **5**(10), 10 (2009)
40.26 D.R. Melmoth, A.L. Finlay, M.J. Morgan, S. Grant: Grasping deficits and adaptations in adults with stereo vision losses, Investig. Ophthalmol. Vis. Sci. **50**(8), 3711–3720 (2009)
40.27 M. Controzzi, C. Cipriani, M.C. Carozza: Design of artificial hands: A review. In: *The Human Hand as an Inspiration for Robot Hand Development*, Springer Tracts in Advanced Robotics, Vol. 95, ed. by R. Balasubramanian, V.J. Santos (Springer, Berlin, Heidlberg 2014) pp. 219–247
40.28 M. Grebenstein: Approaching Human Performance – The Functionality Driven Awiwi Robot Hand, Ph.D. Thesis (ETH, Zürich 2012)
40.29 J. Bae, S. Park, J. Park, M. Baeg, D. Kim, S. Oh: Development of a low cost anthropomorphic robot hand with high capability, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4776–4782
40.30 I. Gaiser, S. Schulz, A. Kargov, H. Klosek, A. Bierbaum, C. Pylatiuk, R. Oberle, T. Werner, T. Asfour, G. Bretthauer, R. Dillmann: A new anthropomorphic robotic hand, 8th IEEE-RAS Int. Conf. Humanoid Robotics (Humanoids) (2008) pp. 418–422
40.31 A.M. Dollar, R.D. Howe: Simple, reliable robotic grasping for human environments, IEEE Int. Conf. Technol. Pract. Robot Appl. (TePRA) (2008) pp. 156–161
40.32 R. Ma, L. Odhner, A. Dollar: A modular, open-source 3-D printed underactuated hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
40.33 F. Ilievski, A. Mazzeo, R.F. Shepherd, X. Chen, G.M. Whitesides: Soft robotics for chemists, Angew. Chem. Int. Ed. **50**(8), 1890–1895 (2011)
40.34 G. Grioli, M. Catalano, E. Silvestro, S. Tono, A. Bicchi: Adaptive synergies: An approach to the design of under-actuated robotic hands, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 1251–1256
40.35 R. Deimel, O. Brock: A novel type of compliant, underactuated robotic hand for dexterous grasping, Proc. Robotics Sci. Syst., Berkeley (2014)
40.36 M.E. Giannaccini, I. Georgilas, I. Horsfield, B.H.P.M. Peiris, A. Lenz, A.G. Pipe, S. Dogramadzi: A variable compliance, soft gripper, Auton. Robots **36**(1/2), 93–107 (2014)
40.37 D. Prattichizzo, M. Malvezzi, M. Gabiccini, A. Bicchi: On the manipulability ellipsoids of underactuated robotic hands with compliance, Robotics Auton. Syst. **60**(3), 337–346 (2012)
40.38 M. Gabiccini, E. Farnioli, A. Bicchi: Grasp analysis tools for synergistic underactuated robotic hands, Int. J. Robotics Res. **32**(13), 1553–1576 (2013)
40.39 Y. Li, A.A. Frank: A moving base robot, Am. Control Conf. 1986 (1986) pp. 1927–1932
40.40 Y. Yamamoto, X. Yun: Coordinating locomotion and manipulation of a mobile manipulator, Proc. 31st IEEE Conf. Decis. Control IEEE (1992) pp. 2643–2648
40.41 Y. Yamamoto, X. Yun: Control of mobile manipulators following a moving surface, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 1–6
40.42 H. Seraji: Configuration control of redundant manipulators: Theory and implementation, IEEE Trans. Robotics Autom. **5**(4), 472–490 (1989)
40.43 H. Seraji, R. Colbaugh: Improved configuration control for redundant robots, J. Robotics Syst. **7**(6), 897–928 (1990)
40.44 H. Seraji: An on-line approach to coordinated mobility and manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 28–35
40.45 H. Seraji: A unified approach to motion control of mobile manipulators, Int. J. Robotics Res. **17**(2), 107–118 (1998)
40.46 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robotics Syst. **13**(11), 755–764 (1996)
40.47 V. Padois, J.-Y. Fourquet, P. Chiron: Kinematic and dynamic model-based control of wheeled mobile manipulators: A unified framework for reactive approaches, Robotica **25**(02), 157–173 (2007)
40.48 E. Papadopoulos, J. Poulakakis: Planning and model-based control for mobile manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 1810–1815
40.49 J. Tan, N. Xi: Unified model approach for planning and control of mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (2001) pp. 3145–3152
40.50 J. Joshi, A.A. Desrochers: Modeling and control of a mobile robot subject to disturbances, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1986) pp. 1508–1513
40.51 N. Hootsmans, S. Dubowsky: Large motion control of mobile manipulators including vehicle

Avidbots Ex1016
Page 1063 of 2154

suspension characteristics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1991) pp. 2336–2341
40.52 K. Liu, F.L. Lewis: Decentralized continuous robust controller for mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 1822–1827
40.53 J.H. Chung, S.A. Velinsky, R.A. Hess: Interaction control of a redundant mobile manipulator, Int. J. Robotics Res. **17**(12), 1302–1309 (1998)
40.54 Y. Yamamoto, X. Yun: Effect of the dynamic interaction on coordinated control of mobile manipulators, IEEE Trans. Robotics Autom. **12**(5), 816–824 (1996)
40.55 S. Lin, A.A. Goldenberg: Robust damping control of mobile manipulators, IEEE Trans. Syst. Man Cybern. B: Cybern **32**(1), 126–132 (2002)
40.56 S. Lin, A.A. Goldenberg: Neural-network control of mobile manipulators, IEEE Trans. Neural Netw. **12**(5), 1121–1133 (2001)
40.57 C.Y. Lee, I.K. Jeong, I.H. Lee, J.J. Lee: Motion control of mobile manipulator based on neural networks and error compensation, Proc. IEEE/RSJ Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4627–4632
40.58 Z. Li, S.S. Ge, A. Ming: Adaptive robust motion/force control of holonomic-constrained nonholonomic mobile manipulators, IEEE Trans. Syst. Man, Cybern. B: Cybern **37**(3), 607–616 (2007)
40.59 Y. Liu, Y. Li: Sliding mode adaptive neural-network control for nonholonomic mobile modular manipulators, J. Intell. Robotics Syst. **44**(3), 203–224 (2005)
40.60 D. Xu, D. Zhao, J. Yi, X. Tan: Trajectory tracking control of omnidirectional wheeled mobile manipulators: Robust neural network-based sliding mode approach, IEEE Trans. Syst. Man Cybern. **39**(3), 788–799 (2009)
40.61 L. Righetti, J. Buchli, M. Mistry, S. Schaal: Inverse dynamics control of floating-base robots with external constraints: A unified view, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1085–1090
40.62 C. Ott, M.A. Roa, G. Hirzinger: Posture and balance control for biped robots based on contact force optimization, 11th IEEE-RAS Int. Conf. Humanoid Robotics (Humanoids) (2011) pp. 26–33
40.63 O. Khatib, L. Sentis, J. Park, J. Warren: Whole-body dynamic behavior and control of human-like robots, Int. J. Humanoid Robotics **1**(01), 29–43 (2004)
40.64 J. Park, O. Khatib: Contact consistent control framework for humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1963–1969
40.65 M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, R. Siegwart: Quadrupedal locomotion using hierarchical operational space control, Int. J. Robotics Res. **33**, 1062 (2014)
40.66 S. McMillan, D.E. Orin, R.B. McGhee: Efficient dynamic simulation of an underwater vehicle with a robotic manipulator, IEEE Trans. Syst. Man Cybern. **25**(8), 1194–1206 (1995)
40.67 T.W. McLain, S.M. Rock, M.J. Lee: Experiments in the coordinated control of an underwater arm/vehicle system. In: *Underwater Robots*, ed. by J. Yuh, T. Ura, G.A. Bekey (Kluwer, Boston 1996) pp. 139–158
40.68 T.J. Tarn, G.A. Shoults, S.P. Yang: A dynamic model of an underwater vehicle with a robotic manipulator using Kane's method. In: *Underwater Robots*, ed. by J. Yuh, T. Ura, G. Bekey (Kluwer, Boston 1996) pp. 195–209
40.69 G. Antonelli, S. Chiaverini: Task-priority redundancy resolution for underwater vehicle-manipulator systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1998) pp. 768–773
40.70 J. Han, J. Park, W.K. Chung: Robust coordinated motion control of an underwater vehicle-manipulator system with minimizing restoring moments, Ocean Eng. **38**(10), 1197–1206 (2011)
40.71 H. Mahesh, J. Yuh, R. Lakshmi: A coordinated control of an underwater vehicle and robotic manipulator, J. Robotics Syst. **8**(3), 339–370 (1991)
40.72 S. Kawamura, N. Sakagami: Analysis on dynamics of underwater robot manipulators based on iterative learning control and time-scale transformation, IEEE Int. Conf. Robotics Autom (ICRA), Vol. 2 (2002) pp. 1088–1094
40.73 G. Antonelli, F. Caccavale, S. Chiaverini: Adaptive tracking control of underwater vehicle-manipulator systems based on the virtual decomposition approach, IEEE Trans. Robotics Autom. **20**(3), 594–602 (2004)
40.74 G. Antonelli, F. Caccavale, S. Chiaverini, L. Villani: Tracking control for underwater vehicle-manipulator systems with velocity estimation, IEEE J. Ocean. Eng. **25**(3), 399–413 (2000)
40.75 M. Orsag, C. Korpela, P. Oh: Modeling and control of mm-uav: Mobile manipulating unmanned aerial vehicle, J. Intell. Robotics Syst. **69**(1–4), 227–240 (2013)
40.76 V. Lippiello, F. Ruggiero: Exploiting redundancy in cartesian impedance control of uavs equipped with a robotic arm, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 3768–3773
40.77 F. Forte, R. Naldi, A. Macchelli, L. Marconi: Impedance control of an aerial manipulator, Am. Control Conf. (ACC) (2002) pp. 3839–3844
40.78 H.-N. Nguyen, D. Lee: Hybrid force/motion control and internal dynamics of quadrotors for tool operation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 3458–3464
40.79 F. Inoue, T. Murakami, K. Ohnishi: A motion control of mobile manipulator with external force, IEEE/ASME Trans. Mechatron. **6**(2), 137–142 (2001)
40.80 Z. Li, J. Gu, A. Ming, C. Xu, M. Shinojo: Intelligent compliant force/motion control of nonholonomic mobile manipulator working on the nonrigid surface, Neural Comput. Appl. **15**(3/4), 204–216 (2006)
40.81 V. Padois, J.-Y. Fourquet, P. Chiron, M. Renaud: On contact transition for nonholonomic mobile manipulators. In: *Experimental Robotics IX*, ed. by O. Khatib, V. Kumar, D. Rus (Springer, Berlin, Heidelberg 2006) pp. 207–216

Avidbots Ex1016
Page 1064 of 2154

40.82 S. Kang, K. Komoriya, K. Yokoi, T. Koutoku, B. Kim, S. Park: Control of impulsive contact force between mobile manipulator and environment using effective mass and damping controls, Int. J. Precis. Eng. Manuf. **11**(5), 697–704 (2010)

40.83 T. Simèon, J.-P. Laumond, J. Cortès, A. Sahbani: Manipulation planning with probabilistic roadmaps, Int. J. Robotics Res. **23**(7/8), 729–746 (2004)

40.84 M. Stilman: Global manipulation planning in robot joint space with task constraints, IEEE Trans. Robotics **26**(3), 576–584 (2010)

40.85 D. Berenson, S.S. Srinivasa, J. Kuffner: Task space regions: A framework for pose-constrained manipulation planning, Int. J. Robotics Res. **30**(12), 1435–1460 (2011)

40.86 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2004)

40.87 D. Berenson, S. Srinivasa, D. Ferguson, J.J. Kuffner: Manipulation planning on constraint manifolds, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 625–632

40.88 M. Rickert, A. Sieverling, O. Brock: Balancing exploration and exploitation in sampling-based motion planning, IEEE Trans. Robotics **30**(6), 1305–1317 (2014)

40.89 K. Harada, E. Yoshida, K. Yokoi (Eds.): *Motion Planning for Humanoid Robots* (Springer, Berlin, Heidelberg 2010)

40.90 W. Choi, J.-C. Latombe: A reactive architecture for planning and executing robot motions with incomplete knowledge, Proc. RSJ/IEEE Int. Conf. Robotics Intell. Syst. (IROS) Osaka, Vol. 1 (1991) pp. 24–29

40.91 S. Quinlan, O. Khatib: Elastic Bands: Connecting path planning and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (1993) pp. 802–807

40.92 O. Brock, O. Khatib: Elastic strips: A framework for motion generation in human environments, Int. J. Robotics Autm. **21**(10), 1031–1052 (2014)

40.93 M. Toussaint: Robot trajectory optimization using approximate inference, Proc. 26th Ann. Int. Conf. Mach. Learn. (2009) pp. 1049–1056

40.94 M. Zucker, N. Ratliff, A.D. Dragan, M. Pivtoraiko, M. Klingensmith, C.M. Dellin, J.A. Bagnell, S.S. Srinivasa: CHOMP: Covariant Hamiltonian optimization for motion planning, Int. J. Robotics Res. **32**(9/10), 1164–1193 (2013)

40.95 M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal: STOMP: Stochastic trajectory optimization for motion planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 4569–4574

40.96 R.R. Burridge, A.A. Rizzi, D.E. Koditschek: Sequential composition of dynamically dexterous robot behaviors, Int. J. Robotics Res. **18**(6), 534–555 (1999)

40.97 Y. Yang, O. Brock: Elastic roadmaps—motion generation for autonomous mobile manipulation, Auton. Robots **28**(1), 113–130 (2010)

40.98 J. Pineau, G. Gordon, S. Thrun: Point-based value iteration: An anytime algorithm for POMDPs, Int. Jt. Conf. Artif. Intell. (IJCAI) (2003) pp. 1025–1032

40.99 H. Kurniawati, D. Hsu, W.S. Lee: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces, Robotics Sci. Syst. (2008)

40.100 J. Van Den Berg, P. Abbeel, K. Goldberg: LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information, Int. J. Robotics Res. **30**(7), 895–913 (2011)

40.101 S. Prentice, N. Roy: The belief roadmap: Efficient planning in belief space by factoring the covariance, Int. J. Robotics Res. **28**(11), 1448–1465 (2009)

40.102 H. Kurniawati, T. Bandyopadhyay, N.M. Patrikalakis: Global motion planning under uncertain motion, sensing, and environment map, Auton. Robots **33**(3), 255–272 (2012)

40.103 L.P. Kaelbling, T. Lozano-Pérez: Integrated task and motion planning in belief space, Int. J. Robotics Res. **32**(9/10), 1194–1227 (2013)

40.104 A. Sieverling, N. Kuhnen, O. Brock: Sensor-based, task-constrained motion generation under uncertainty, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.105 J. Weston, F. Ratle, R. Collobert: Deep learning via semi-supervised embedding, Proc. 25th Int. Conf. Mach. Learn. (ICML) (2008)

40.106 T.G. Dietterich, P. Domingos, L. Getoor, S. Muggleton, P. Tadepalli: Structured machine learning: The next ten years, Mach. Learn. **73**, 3–23 (2008)

40.107 R. Douglas, T. Sejnowski: *Future Challenges for the Science and Engineering of Learning*, Final NSF Workshop Report (NSF, Arlington County 2008)

40.108 J. Lafferty, A. McCallum, F. Pereira: Conditional random fields: Probabilistic models for segmenting and labeling sequence data, Int. Conf. Mach. Learn. (ICML) (2001) pp. 282–289

40.109 Y. Altun, I. Tsochantaridis, T. Hofmann: Hidden Markov support vector machines, Proc. 20th Int. Conf. Mach. Learn. (2003) pp. 3–10

40.110 A. Saxena, M. Sun, A.Y. Ng: Make 3-D: Depth perception from a single still image, Proc. 23rd AAAI Conf. Artif. Intell. (2008) pp. 1571–1576

40.111 A. Saxena, J. Driemeyer, A.Y. Ng: Robotic grasping of novel objects using vision, Int. J. Robotics Res. **27**(2), 157–173 (2008)

40.112 Y. Jiang, M. Lim, C. Zheng, A. Saxena: Learning to place new objects in a scene, Int. J. Robotics Res. **31**(9), 1021–1043 (2012)

40.113 Y. Jiang, A. Saxena: Modeling and control of a mobile robot subject to disturbances, Proc. Robotics Sci. Syst., Berkeley (2014)

40.114 R.B. Rusu, N. Blodow, M. Beetz: Fast point feature histograms (fpfh) for 3-D registration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 3212–3217

40.115 K. Lai, L. Bo, D. Fox: Unsupervised feature learning for 3-D scene labeling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.116 F. Stulp, E. Theodorou, J. Buchli, S. Schaal: Learning to grasp under uncertainty, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5703–5708

Avidbots Ex1016
Page 1065 of 2154

40.117 M. Cakmak, A.L. Thomaz: Designing robot learners that ask good questions, Proc. Int. Conf. Hum.-Robot Interact. (HRI) (2012)

40.118 N. Jetchev, M. Toussaint: Fast motion planning from experience: Trajectory prediction for speeding up movement generation, Auton. Robots **34**, 111–127 (2013)

40.119 N. Jetchev, M. Toussaint: Discovering relevant task spaces using inverse feedback control, Auton. Robots **37**, 169–189 (2014)

40.120 C. Eppner, O. Brock: Grasping unknown objects by exploiting shape adaptability and environmental constraints, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013)

40.121 M. Toussaint, N. Ratliff, J. Bohg, L. Righetti, P. Englert, S. Schaal: Dual execution of optimized contact interaction trajectories, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014)

40.122 S. Niekum, S. Osentoski, G. Konidaris, A.G. Barto: Learning and generalization of complex tasks from unstructured demonstrations, IEEE/RSJ Int. Conf. Intell. Robotics Systems (IROS) (2012) pp. 5239–5246

40.123 S. Alexandrova, M. Cakmak, K. Hsiao, L. Takayama: Robot programming by demonstration with interactive action visualizations, Proc. Robotics Sci. Syst. (2014)

40.124 M.M. Chun, R. Marois: The dark side of visual attention, Curr. Opin. Neurobiol. **12**(2), 184–189 (2002)

40.125 D. Marr: *Vision: A Computational Investigation Into the Human Representation and Processing of Visual Information* (Henry Holt, New York 1982)

40.126 R.A. Newcombe, S.J. Lovegrove, A.J. Davison: Dtam: Dense tracking and mapping in real-time, IEEE Int. Conf. Comput. Vis. (ICCV) (2011) pp. 2320–2327

40.127 J. Engel, T. Schöps, D. Cremers: LSD-SLAM: Large-scale direct monocular SLAM, Eur. Conf. Comput. Vis. (ECCV) (2014)

40.128 P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox: Rgb-d mapping: Using kinect-style depth cameras for dense 3-D modeling of indoor environments, Int. J. Robotics Res. **31**(5), 647–663 (2012)

40.129 J. Stückler, S. Behnke: Multi-resolution surfel maps for efficient dense 3-D modeling and tracking, J. Vis. Commun. Image Represent. **25**(1), 137–147 (2014)

40.130 K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, W. Burgard: Octomap: A probabilistic, flexible, and compact 3-D map representation for robotic systems, Proc. ICRA 2010 Workshop Best Practice 3-D Percept. Model. Mob. Manip., Vol. 2 (2010)

40.131 A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard: Octomap: An efficient probabilistic 3-D mapping framework based on octrees, Auton. Robots **34**(3), 189–206 (2013)

40.132 J.P. Saarinen, H. Andreasson, T. Stoyanov, A.J. Lilienthal: 3-D normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments, Int. J. Robotics Res. **32**(14), 1627–1644 (2013)

40.133 D. Droeschel, J. Stückler, S. Behnke: Local multi-resolution representation for 6d motion estimation and mapping with a continuously rotating 3-D laser scanner, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.134 D. Droeschel, J. Stückler, S. Behnke: Local multiresolution surfel grids for MAV motion estimation and 3-D mapping, Proc. Int. Conf. Intell. Auton. Syst. (IAS) (2014)

40.135 M. McElhone, J. Stuckler, S. Behnke: Joint detection and pose tracking of multi-resolution surfel models in RGB-D, IEEE Eur. Conf. Mobile Robots (ECMR) (2013) pp. 131–137

40.136 J. Stückler, B. Waldvogel, H. Schulz, S. Behnke: Dense real-time mapping of object-class semantics from RGB-D video, J. Real-Time Image Process. **8**, 1–11 (2014)

40.137 B. Lau, C. Sprunk, W. Burgard: Efficient grid-based spatial representations for robot navigation in dynamic environments, Robotics Auton. Syst. **61**(10), 1116–1130 (2013)

40.138 R. Bis, H. Peng, G. Ulsoy: Velocity occupancy space: Robot navigation and moving obstacle avoidance with sensor uncertainty, ASME Dyn. Syst. Control Conf. (2009) pp. 363–370

40.139 A.N. Ghomsheh, A. Talebpour: A new method for indoor-outdoor image classification using color correlated temperature, Int. J. Image Process **6**(3), 167–181 (2012)

40.140 A. Payne, S. Singh: Indoor vs. outdoor scene classification in digital photographs, Pattern Recogn. **38**(10), 1533–1545 (2005)

40.141 J. Collier, A. Ramirez-Serrano: Environment classification for indoor/outdoor robotic mapping, IEEE Can. Conf. Comput. Robot Vis. (CRV) (2009) pp. 276–283

40.142 J. Aloimonos, I. Weiss, A. Bandyopadhyay: Active vision, Int. J. Comput. Vis. **1**(4), 333–356 (1988)

40.143 R. Bajcsy: Active perception, Proceedings IEEE **76**(8), 996–1006 (1988)

40.144 A. Blake, A. Yuille: *Active Vision* (MIT, Cambridge 1992)

40.145 A. Petrovskaya, O. Khatib: Global localization of objects via touch, IEEE Trans. Robotics **27**(3), 569–585 (2011)

40.146 P. Hebert, T. Howard, N. Hudson, J. Ma, J.W. Burdick: The next best touch for model-based localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 99–106

40.147 H. Lee, J. Park: An active sensing strategy for contact location without tactile sensors using robot geometry and kinematics, Auton. Robots **36**(1/2), 109–121 (2014)

40.148 M.J. Kim, M. Choi, Y.B. Kim, F. Liu, H. Moon, J.C. Koo, H.R. Choi: Exploration of unknown object by active touch of robot hand, Int. J. Control Autom. Syst. **12**(2), 406–414 (2014)

40.149 L. Chang, J.R. Smith, D. Fox: Interactive singulation of objects from a pile, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

Avidbots Ex1016
Page 1066 of 2154

40.150 M. Gupta, G.S. Sukhatme: Using manipulation primitives for brick sorting in clutter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

40.151 D. Katz, M. Kazemi, J.A. Bagnell, A. Stentz: Autonomous pile clearing using interactive perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

40.152 M.R. Dogar, M.C. Koval, A. Tallavajhula, S.S. Srinivasa: Object search by manipulation, Auton. Robots **36**(1/2), 153–167 (2014)

40.153 R. Martín Martín, O. Brock: Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014)

40.154 D. Katz, O. Brock: Manipulating articulated objects with interactive perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 272–277

40.155 D. Katz, A. Orthey, O. Brock: Interactive perception of articulated objects, Int. Symp. Exp. Robotics (2010)

40.156 D. Katz, A. Orthey, O. Brock: Interactive perception of articulated objects, Springer Tract. Adv. Robotics **79**, 301–315 (2014)

40.157 D. Katz, M. Kazemi, J.A. Bagnell, A. Stentz: Interactive segmentation, tracking, and kinematic modeling of unknown articulated objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 5003–5010

40.158 X. Huang, I. Walker, S. Birchfield: Occlusion-aware multi-view reconstruction of articulated objects for manipulation, Robotics Auton. Syst. **63**(4), 497–505 (2014)

40.159 S. Pillai, M. Walter, S. Teller: Learning articulated motions from visual demonstration, Proc. Robotics Sci. Syst. (2014)

40.160 J. Sturm, K. Konolige, C. Stachniss, W. Burgard: Vision-based detection for learning articulation models of cabinet doors and drawers in household environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 362–368

40.161 J. Sturm, A. Jain, C. Stachniss, C. Kemp, W. Burgard: Operating articulated objects based on experience, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

40.162 J. Sinapov, T. Bergquist, C. Schenck, U. Ohiri, S. Griffith, A. Stoytchev: Interactive object recognition using proprioceptive and auditory feedback, Int. J. Robotics Res. **30**(10), 1250–1262 (2011)

40.163 D. Katz, Y. Pyuro, O. Brock: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation, Proc. Robotics Sci. Syst., Zurich (2008) pp. 254–261

40.164 S. Otte, J. Kulick, M. Toussaint, O. Brock: Entropy-based strategies for physical exploration of the environment's degrees of freedom, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014)

40.165 S. Höfer, T. Lang, O. Brock: Extracting kinematic background knowledge from interactions using task-sensitive relational learning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.166 J.J. Gibson: The theory of affordances. In: *Perceiving, Acting, and Knowing*, ed. by R. Shaw, J. Bransford (Lawrence Erlbaum, Hilldale 1977)

40.167 E. Şahin, M. Çakmak, M.R. Doğar, E. Uğur, G. Üçoluk: To afford or not to afford: A new formalization of affordances toward affordance-based robot control, Adapt. Behav. **15**(4), 447–472 (2007)

40.168 L. Montesano, M. Lopes, A. Bernardino, J. Santos-Victor: Learning object affordances: From sensory–motor coordination to imitation, IEEE Trans. Robotics **24**(1), 15–26 (2008)

40.169 L. Paletta, G. Fritz, F. Kintzler, J. Irran, G. Dorffner: Learning to perceive affordances in a framework of developmental embodied cognition, IEEE 6th Int. Conf. IEEE Dev. Learn. (ICDL) (2007) pp. 110–115

40.170 P. Sequeira, M. Vala, A. Paiva: What can i do with this?: Finding possible interactions between characters and objects, Proc. 6th ACM Int. Jt. Conf. Auton. Agents Multiagent Syst. (2007) p. 5

40.171 B. Moldovan, M. van Otterlo, P. Moreno, J. Santos-Victor, L. De Raedt: Statistical relational learning of object affordances for robotic manipulation, Latest Adv. Inductive Logic Program. (2012) p. 6

40.172 C. de Granville, J. Southerland, A.H. Fagg: Learning grasp affordances through human demonstration, Proc. Int. Conf. Dev. Learn. (ICDL06) (2006)

40.173 L. Montesano, M. Lopes: Learning grasping affordances from local visual descriptors, IEEE 8th Int. Conf. Dev. Learn. (ICDL) (2009) pp. 1–6

40.174 D. Kraft, R. Detry, N. Pugeault, E. Baseski, J. Piater, N. Krüger: Learning objects and grasp affordances through autonomous exploration, Lect. Notes Comput. Sci. **5815**, 235–244 (2009)

40.175 R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, J. Piater: Learning grasp affordance densities, Paladyn **2**(1), 1–17 (2011)

40.176 T. Gollisch, M. Meister: Eye smarter than scientists believed: Neural computations in circuits of the retina, Neuron **65**(2), 150–164 (2010)

Avidbots Ex1016
Page 1067 of 2154

Multimedia Contents



41. Active Manipulation for Perception

Anna Petrovskaya, Kaijen Hsiao

This chapter covers perceptual methods in which manipulation is an integral part of perception. These methods face special challenges due to data sparsity and high costs of sensing actions. However, they can also succeed where other perceptual methods fail, for example, in poor-visibility conditions or for learning the physical properties of a scene.

The chapter focuses on specialized methods that have been developed for object localization, inference, planning, recognition, and modeling in active manipulation approaches. We conclude with a discussion of real-life applications and directions for future research.

41.1 **Perception via Manipulation** 1037

41.2 **Object Localization** 1038
41.2.1 Problem Evolution 1038
41.2.2 Bayesian Framework 1039
41.2.3 Inference 1041
41.2.4 Advanced Inference Methods 1043
41.2.5 Planning 1045

41.3 **Learning About an Object** 1049
41.3.1 Shape of Rigid Objects 1049
41.3.2 Articulated Objects 1051
41.3.3 Deformable Objects 1052
41.3.4 Contact Estimation 1053
41.3.5 Physical Properties 1054

41.4 **Recognition** 1054
41.4.1 Object Shape Matching 1054
41.4.2 Statistical Pattern Recognition 1055
41.4.3 Recognition by Material Properties 1056
41.4.4 Combining Visual and Tactile Sensing 1056

41.5 **Conclusions** 1057
41.5.1 Real-Life Applications 1057
41.5.2 Future Directions of Research 1057
41.5.3 Further Reading 1057

Video-References 1058

References 1058

41.1 Perception via Manipulation

In this chapter, we focus on perceptual methods that rely on manipulation. As we will see, sensing-*via*-manipulation is a complex and highly intertwined process, in which the results of perception are used for manipulation and manipulation is used to gather additional data. Despite this complexity, there are three main reasons to use manipulation either together or instead of the more commonly used vision sensors.

First, manipulation can be used to sense in poor-visibility conditions: for example, sensing in muddy water or working with transparent objects. In fact, due to its high accuracy and ability to sense any hard surface regardless of its optical properties, contact sensing is widely used for accurate localization of parts in manufacturing. Second, manipulation is useful for determining properties that require physical interaction, such as stiffness, mass, or the physical relationship of parts. Third, if the actual goal is to manipulate the object, we might as well use data gathered from manipulation attempts to improve both perception and subsequent attempts at manipulation.

Sensing-*via*-manipulation also faces significant challenges. Unlike vision sensors, which provide a whole-scene view in a single snapshot, contact sensors are inherently local, providing information only about a very small area of the sensed surface at any given time. In order to gather additional data, the manipu-

lator has to be moved into a different position, which is a time-consuming task. Hence, unlike vision-based perception, contact-based methods have to cope with very sparse data and very low data acquisition rates. Moreover, contact sensing disturbs the scene. While in some situations this can be desirable, it also means that each sensing action can increase uncertainty. In fact, one can easily end up in a situation where the information gained via sensing is less than the information lost due to scene disturbance.

Due to these challenges, specialized perceptual methods had to be developed for perception-via-manipulation. Inference methods have been developed to extract the most information from the scarce data and to cope with cases in which the data are insufficient to fully solve the problem (i. e., under-constrained scenarios). Planning methods have been developed to make the most efficient sensing decisions based on the little data available and taking into account time, energy, and uncertainty costs of sensing actions. Modeling methods have been developed to build the most accurate models from small amounts of data.

In this chapter, the material is organized by perceptual goal: localizing an object (Sect. 41.2), learning about an object (Sect. 41.3), and recognizing an object (Sect. 41.4). Since the chapter topic is broad, to keep it concise, we provide a tutorial-level discussion of object localization methods, but give a birds-eye view of the methods for the other two perceptual goals. However, as we point out along the way, many object localization methods can be reused for the other two perceptual goals.

41.2 Object Localization

Tactile object localization is the problem of estimating the object's pose – including position and orientation – based on a set of data obtained by touching the object. A prior geometric model of the object is assumed to be known and the sensing is performed using some type of contact sensor (wrist force/torque sensor, tactile array, fingertip force sensor, etc.). The problem is typically restricted to rigid objects, which can be stationary or moving.

Tactile object localization involves several components, including manipulator control methods, modeling, inference, and planning. During data collection, some form of sensorimotor control is used for each sensing motion of the manipulator. For information on manipulator control, the reader is encouraged to reference Chaps. 8 and 9. With few exceptions, the sensing motions tend to be poking motions rather than following the surface. This is due to the fact that robots are usually unable to sense with all parts of the manipulator, and, hence, the poking motions are chosen to minimize the possibility of accidental non-sensed contact.

Due to the properties of sensing-via-contact, special methods for modeling, inference, and planning had to be developed for tactile localization. Modeling choices include not only how to model the object itself, but also models of the sensing process and models of possible object motion. We discuss models of sensing and motion in Sect. 41.2.2, but leave an in-depth discussion of object modeling techniques until Sect. 41.3. We also cover specialized inference and planning methods in Sect. 41.2.3 and 41.2.5, respectively.

41.2.1 Problem Evolution

Attempts to solve the tactile object localization date back to early 80s. Over the years, the scope of the problem as well as methods used to solve it have evolved as we detail below.

Early Methods

Early methods for tactile object localization generally ignore the sensing process uncertainties and focus on finding a single hypothesis that best fits the measurements. For example, *Gaston* and *Lozano-Perez* used interpretation trees to efficiently find the best match for 3 DOF (degrees of freedom) object localization [41.1]. *Grimson* and *Lozano-Perez* extended the approach to 6 DOF [41.2]. *Faugeras* and *Hebert* used least squares to perform geometrical matching between primitive surfaces [41.3]. *Shekhar* et al. solved systems of weighted linear equations to localize an object held in a robotic hand [41.4]. Several methods use geometric constraints together with kinematic and dynamic equations to estimate or constrain the pose of known planar objects in plane by pushing them with a finger [41.5] or parallel-jaw gripper [41.6], or by tilting them in a tray [41.7].

Workpiece Localization

Single hypothesis methods are also widely used to solve the workpiece localization problem in manufacturing applications for dimensional inspection [41.8], machining [41.9], and robotic assembly [41.10]. In these applications, the measurements are taken by a coordinate measurement machine (CMM) [41.11] or by on-machine sensors [41.12]. Workpiece localization makes

a number of restrictive assumptions that make it inapplicable to autonomous robot operation in unstructured environments. One important restriction is that there is a known correspondence between each measured data point and a point or patch on the object surface (called home point or home surface, respectively) [41.13]. In semiautomated settings, the correspondence assumption is satisfied by having a human direct the robot to specific locations on the object. In fully automated settings, the object is placed on the measurement table with low uncertainty to make sure each data point lands near the corresponding home point.

Further restrictions include assumptions that the data are sufficient to fully constrain the object, the object does not move, and there are no unmodeled effects (e.g., vibration, deformation, or temperature variation). All of these parameters are carefully controlled for in structured manufacturing environments.

The workpiece localization problem is usually solved in least squares form using iterative optimization methods, including the Hong-Tan method [41.14], the Variational method [41.15], and the *Menq* method [41.16]. Since these methods are prone to getting trapped in local minima, low initial uncertainty is usually assumed to make sure the optimization algorithm is initialized near the solution. Some attempts have been made to solve the global localization problem by rerunning the optimization algorithm multiple times from prespecified and random initial points [41.17]. Recent work has focused on careful selection of the home points to improve localization results [41.18–20] and on improving localization efficiency for complex home surfaces [41.21, 22].

Bayesian Methods

In the last decade, there has been increased interest in Bayesian state estimation for the tactile object localization problem [41.23–27]. These methods estimate the probability distribution over all possible states (the belief), which captures the uncertainty resulting from noisy sensors, inaccurate object models, and other effects present during the sensing process. Thus, estimation of the belief enables planning algorithms that are resilient to the uncertainties of the real world. Unlike workpiece localization, these methods do not assume known correspondences. In contrast to single hypothesis or set-based methods, belief estimation methods allow us to better describe and track the relative likelihoods of hypotheses in the under-constrained scenario, in which the data are insufficient to fully localize the object. These methods can also work with moving objects and answer important questions such as: *have we localized the object completely?* and *where is the best place to sense next?*.

41.2.2 Bayesian Framework

All Bayesian methods share a similar framework. We start with a general definition of a Bayesian problem (not necessarily tactile object localization), and then, explain how this formulation can be applied to tactile localization.

General Bayesian Problem

For a general Bayesian problem, the goal is to infer the state X of a system based on a set of sensor measurements $\mathcal{D} := \{D_k\}$. Due to uncertainty, this information is best captured as a probability distribution

$$bel(X) := p(X|\mathcal{D}) \,, \tag{41.1}$$

called the posterior distribution or the Bayesian belief. Figure 41.1a shows a Bayesian network representing all the random variables involved and all the relationships between them.

In a dynamic Bayesian system, the state changes over time, which is assumed to be discretized into small time intervals (Fig. 41.1b). The system is assumed to evolve as a Markov process with unobserved states. The goal is to estimate the belief at time t

$$bel_t(X_t) := p(X_t|\mathcal{D}_1, \ldots, \mathcal{D}_t) \,. \tag{41.2}$$

The behavior of the system is described via two probabilistic laws: (i) the measurement model $p(\mathcal{D}|X)$ captures how the sensor measurements are obtained and (ii) the dynamics model $p(X_t|X_{t-1})$ captures how the system evolves between time steps.

For brevity, it is convenient to drop the arguments in $bel(X)$ and $bel_t(X_t)$, and simply write bel and bel_t, but these two beliefs should always be understood as functions of X and X_t, respectively.

Tactile Localization in Bayesian Form

Tactile object localization can be formulated as an instance of the general Bayesian problem. Here, the

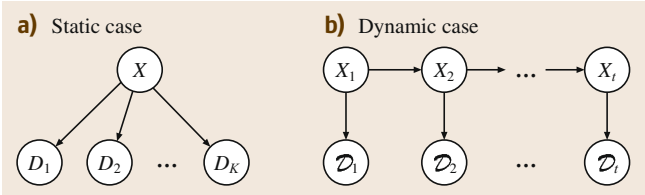


Fig.41.1a,b Bayesian network representation of the relationships between the random variables involved in a general Bayesian problem. The directional arrows are read as *causes*. **(a)** In the static case, a single unknown state X causes a collection of measurements $\{D_k\}$. **(b)** In the dynamic case, the state X_t changes over time and at each time step causes a set of measurements $\mathcal{D}_t$

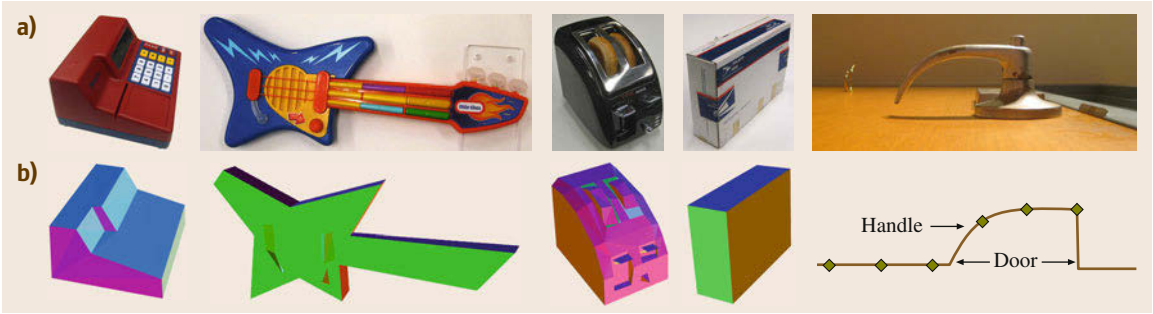


Fig.41.2a,b Examples of objects **(a)** and their polygonal mesh models **(b)**. Five objects are shown: cash register, toy guitar, toaster, box, and door handle. The first three models were constructed by collecting surface points with the robot's end effector. The last two were constructed from hand measurements made with a ruler. The door handle model is 2-D. Model complexity ranges from 6 faces (for the box) to over 100 faces (for the toaster) (after [41.28])

robot needs to determine the pose X of a known object $\mathcal{O}$ based on a set of tactile measurements $\mathcal{D}$. The state is the 6 DOF pose of the object – including position and orientation – in the manipulator coordinate frame. A number of state parameterizations are possible, including matrix representations, quaternions, Euler angles, and Rodrigues angles, to name a few. For our discussion here, assume that the state $X := (x, y, z, \alpha, \beta, \gamma)$, where (x, y, z) is the position and (α, β, γ) are orientation angles in Euler representation.

The measurements $\mathcal{D} := \{D_k\}$ are obtained by touching the object with the robot's end effector. The end effector can be a single probe, a gripper, or a hand. Thus, it may be able to sense multiple contacts D_k simultaneously, for example, in a single grasping attempt. We will assume that for each touch D_k, the robot is able to measure the contact point and also possibly sense the normal of the surface (perhaps as the direction of reaction force). Under these assumptions, each measurement $D_k := (D_k^{\text{pos}}, D_k^{\text{nor}})$ consists of the measured Cartesian position of the contact point D_k^{pos} and the measured surface normal D_k^{nor}. If measurements of surface normals are not available, then $D_k := D_k^{\text{pos}}$.

Measurement Model

In order to interpret tactile measurements, we need to define a model of the object and a model of the sensing process. For now, assume that the object is modeled as a polygonal mesh, which could be derived from a CAD model or a 3-D scan of the object, or built using contact sensors. Examples of objects and their polygonal mesh representations are shown in Fig. 41.2. Other types of object models can also be used. We provide an in-depth look at object modeling techniques in Sect. 41.3.

Typically, the individual measurements D_k in a data set $\mathcal{D}$ are considered independent of each other given the state X. Then, the measurement model factors over the measurements

$$p(\mathcal{D}|X) = \prod_k p(D_k|X) \,. \tag{41.3}$$

Sampled Models. Early Bayesian tactile localization work used sampled measurement models. For example, *Gadeyne* and *Bruyninckx* used numerical integration to compute the measurement model for a box and stored it in a look-up table for fast access [41.23]. *Chhatpar* and *Branicky* sampled the object surface by repeatedly touching it with the robot's end-effector to compute the measurement probabilities (Fig. 41.3) [41.24].

Proximity Model. One common model for the interpretation of tactile measurements is the proximity measurement model [41.29]. In this model, the measurements are considered to be independent of each other, with both position and normal components corrupted by Gaussian noise. For each measurement, the probability depends on the distance between the measurement and the object surface (hence the name *proximity*).

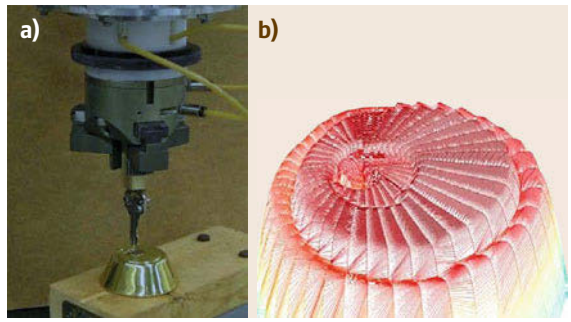


Fig.41.3a,b Sampled measurement model for a key lock. **(a)** The robot exploring the object with a key. **(b)** The resulting lock-key contact C-space (after [41.24])

Since the measurements contain both contact coordinates and surface normals, this distance is taken in the 6-D space of coordinates and normals (i. e., in the measurement space). Let $\hat{\mathcal{O}}$ be a representation of the object in this 6-D space. Let $\hat{o} := (\hat{o}^{\text{pos}}, \hat{o}^{\text{nor}})$ be a point on the object surface, and D be a measurement. Define $d_{\text{M}}(\hat{o}, D)$ to be the Mahalanobis distance between $\hat{o}$ and D

$$d_{\text{M}}(\hat{o}, D) := \sqrt{\frac{||\hat{o}^{\text{pos}} - D^{\text{pos}}||^2}{\sigma_{\text{pos}}^2} + \frac{||\hat{o}^{\text{nor}} - D^{\text{nor}}||^2}{\sigma_{\text{nor}}^2}} , \tag{41.4}$$

where σ_{pos}^2 and σ_{nor}^2 are Gaussian noise variances of position and normal measurement components, respectively. In the case of a sensor that only measures position (and not surface normal), only the first summand is used inside the square root. The distance between a measurement D and the entire object $\hat{\mathcal{O}}$ is obtained by minimizing the Mahalonobis distance over all object points $\hat{o}$

$$d_{\text{M}}(\hat{\mathcal{O}}, D) := \min_{\hat{o} \in \hat{\mathcal{O}}} d_{\text{M}}(\hat{o}, D) . \tag{41.5}$$

Let $\hat{\mathcal{O}}_X$ denote the object in state X. Then, the measurement model is computed as

$$p(\mathcal{D}|X) = \eta \exp\left[-\frac{1}{2}\sum_k d_{\text{M}}^2(\hat{\mathcal{O}}_X, D_k)\right] . \tag{41.6}$$

In the above equation and throughout the chapter, η denotes the normalization constant, whose value is such that the expression integrates to 1. VIDEO 723 shows the proximity model being used to localize a door handle.

Integrated Proximity Model. A variation of the proximity model is called the integrated proximity model. Instead of assuming that the closest point on the object caused the measurement, it considers the contribution from all surface points to the probability of the measurement [41.25]. This is a much more complex model that in general can not be computed efficiently. Moreover, for an unbiased application of this model, we need to compute a prior over all surface points, i. e., how likely each surface point is to cause a measurement. This prior is usually nonuniform and highly dependent on the object shape, the manipulator shape, and the probing motions. Nevertheless, in some cases this model can be beneficial as it is more expressive than the proximity model.

Negative Information. The models we have described so far do not take into account negative information, i. e., information about the absence rather than presence of measurements. This includes information that the robot was able to move through some parts of space without making contact with the object. Negative information is very useful for active exploration strategies and has been taken into account by [41.26] and [41.25]. Although adding negative information makes the belief more complex (e.g., discontinuous), in most cases, it can be superimposed on top of the belief computed using one of the above measurement models.

Dynamics Model

Free-standing objects can move during probing, and hence, a dynamics model is needed to describe this process. In most situations, little is known about possible object motions, and thus, a simple Gaussian model is assumed. Hence, $p(X_t|X_{t-1})$ is a Gaussian with mean at X_{t-1} and variances σ_{met}^2 and σ_{ang}^2 along metric and angular axes respectively. If additional properties of object motion are known, then a more informative dynamics model can be used. For example, if we know the robot shape and the magnitude and direction of the contact force, we can use Newtonian dynamics to describe object motion. This motion may be further restricted, if it is known that the object is sliding on a specific surface.

41.2.3 Inference

Once the models are defined and the sensor data are obtained, the next step is to estimate the resulting probabilistic belief. This process is called inference and can be viewed as numerical estimation of a real-valued function over a multidimensional space. We can distinguish two cases: a static case where the object is rigidly fixed and a dynamic case where the object can move. The dynamic case is solved recursively at each time step using a Bayesian filter. The static case can be solved either recursively or in a single step by combining all the data into a single batch. The other important differentiator is the amount of initial uncertainty. In low uncertainty problems, the pose of the object is approximately known and only needs to be updated based on the latest data. This type of problem usually arises during tracking of a moving object, in which case it is known as pose tracking. In global uncertainty problems, the object can be anywhere within a large volume of space and can have arbitrary orientation. This case is often referred to as global localization. Global localization problems arise for static objects, when the pose of the object is unknown, or as the initial step in pose tracking of a dynamic object. It is also the fallback

whenever the pose tracking method fails and uncertainty about the object pose becomes large.

The belief tends to be a highly complex function with many local extrema. For this reason, parametric methods (such as Kalman filters) do not do well for this problem. Instead, the problem is usually solved using nonparametric methods. The complexity of the belief is caused directly by properties of the world, the sensors, and models thereof. For more insight into causes of belief complexity and appropriate estimation methods, see the discussion of belief roughness in [41.28, Chap. 1].

We start by describing basic nonparametric methods, which are efficient enough to solve problems with up to three degrees of freedom (DOFs). Then, in Sect. 41.2.4, we describe several advanced methods that have been developed to solve the full 6 DOF problem in real-time.

Basic Nonparametric Methods

Nonparametric methods typically approximate the belief by points. There are two types of nonparametric methods: deterministic and Monte Carlo (i. e., nondeterministic). The most common deterministic methods are grids and histogram filters (HF). For these methods, the points are arranged in a grid pattern with each point representing a grid cell. The most common Monte Carlo methods are importance sampling (IS) and particle filters (PF). For Monte Carlo methods, the points are sampled randomly from the state space and called samples or particles.

Static Case

Via Bayes rule, the belief $bel(X)$ can be shown to be proportional to $p(\mathcal{D}|X)p(X)$. The first factor is the measurement model. The second factor, $\overline{bel}(X) := p(X)$, is called the Bayesian prior, which represents our belief about X before obtaining measurements $\mathcal{D}$. Hence, with this notation, we can write

$$bel = \eta\, p(\mathcal{D}|X)\, \overline{bel}\,. \tag{41.7}$$

In the most common case, where it is only known that the object is located within some bounded region of space, a uniform prior is most appropriate. In this case, the belief bel is proportional to the measurement model and the equation (41.7) simplifies to

$$bel = \eta\, p(\mathcal{D}|X)\,. \tag{41.8}$$

However, if an approximate object pose is known ahead of time (e.g., an estimate produced from a vision sensor), then it is common to use a Gaussian prior of some covariance around the approximate pose.

Grid methods compute the unnormalized belief $p(\mathcal{D}|X)\,\overline{bel}$ at the center of each grid cell and then normalize over all the grid cells to obtain an estimate of the belief bel.

Monte Carlo methods use importance sampling. For uniform priors, the particles are sampled uniformly from the state space. For more complex priors, the particles have to be sampled from the prior distribution $\overline{bel}$. The importance weight for each particle is set to the measurement model $p(\mathcal{D}|X)$ and the entire set of importance weights is normalized so that the weights add up to 1.

Dynamic Case

Given the measurement model $p(\mathcal{D}|X)$, the dynamics model $p(X_t|X_{t-1})$, and measurements $\{\mathcal{D}_1, \ldots, \mathcal{D}_t\}$ up to time t, the belief can be computed recursively using a Bayesian filter algorithm, which relies on the Bayesian recursion equation

$$bel_t = \eta\, p(\mathcal{D}_t|X_t)\, \overline{bel}_t\,. \tag{41.9}$$

To compute (41.9), the Bayesian filter alternates two steps: (a) the dynamics update, which computes the prior

$$\overline{bel}_t = \eta \int p\,(X_t|X_{t-1})\, bel_{t-1}\, \mathrm{d}X_{t-1}\,, \tag{41.10}$$

and (b) the measurement update, which computes the measurement model $p(\mathcal{D}_t|X_t)$.

A grid-based implementation of the Bayesian filter is called a histogram filter. Histogram filters compute (41.9) for each grid cell separately. At the start of time step t, the grid cell stores its estimate of the belief bel_{t-1} from the previous time step. During the dynamics update, the value of the prior $\overline{bel}_t$ at this grid cell is computed using (41.10), where the integral is replaced by a summation over all grid cells. During the measurement update, the measurement model is computed at the center of the grid cell and multiplied by the value of the grid cell (i. e., the prior). Then, values of all grid cells are normalized so they add up to 1. Since during dynamics update, a summation over all grid cells has to be carried out for each grid cell, the computational complexity of histogram filters is quadratic in the number of grid cells.

A Monte Carlo implementation of the Bayesian filter is called a particle filter. The particle set representing the prior $\overline{bel}_t$ is produced during the dynamics update as follows. The particles are resampled (see [41.30] for a tutorial on resampling) from the belief at the previous time step, bel_{t-1}, and then moved according to the dynamics model with the addition of some motion noise. This produces a set of particles representing the prior $\overline{bel}_t$. The measurement update is performed using importance sampling as in the static case above. Note that

Avidbots Ex1016
Page 1073 of 2154

unlike for the histogram filters, there is no per-particle summation during the dynamics update, and thus, the computational complexity of particle filters is linear in the number of particles.

41.2.4 Advanced Inference Methods

Although suitable for some problems, basic nonparametric methods have a number of shortcomings. There are two main issues.

1. While a rigid body positioned in space has 6 DOFs, the basic methods can only be used for localization with up to 3 DOFs. Beyond 3 DOFs, these methods become computationally prohibitive due to exponential growth in the number of particles (or grid cells) required. The reason for the exponential blow-up is as follows. In order to localize the object, we need to find its most likely poses, i. e., the peaks of bel_t. These peaks (also called modes) are very narrow, and, hence, the space needs to be densely populated with particles in order to locate them. The number of particles required to achieve the same density per unit volume goes up exponentially with the number of DOFs. Hence, the computational complexity of nonparametric methods also grows exponentially with the number of DOFs. This is known as the curse of dimensionality.
2. Another important issue is that the performance of basic nonparametric methods actually *degrades with increase in sensor accuracy*. This problem is especially pronounced in particle filters. Although it may seem very unintuitive at first, the reason is simple. The more accurate the sensor the narrower the peaks of the belief. Thus, more and more particles are required to find these peaks. We can call this problem curse of accurate sensor.

Several more advanced methods have been developed to combat the shortcomings of basic methods. Below, we detail one such method, called scaling series, and briefly discuss a few others.

Scaling Series

Scaling series is an inference algorithm designed to scale better to higher dimensional problems than the basic methods [41.28, Chap. 2]. In particular, it is capable of solving the full 6 DOF localization problem. Moreover, unlike for basic methods, the accuracy of scaling series *improves with increase in sensor accuracy*. For this reason, this algorithm may be preferable to basic methods even in low-dimensional problems.

Scaling series represents the belief by broad particles. Each particle is thought of as a representative for an entire δ-neighborhood, that is, the volume of space of radius δ surrounding the sample point. Instead of fixing the total number of particles (as in the basic methods), scaling series adjusts this number as needed to obtain good coverage of main peaks.

Peak width can be controlled using annealing, which means that for a given temperature τ, the measurement model is raised to the power of $1/\tau$. Hence, for $\tau = 1$, the original measurement model is obtained, whereas for $\tau > 1$ the measurement model is *heated-up*. Annealing broadens the peaks of bel_t, making them easier to find. However, it also increases ambiguity and decreases accuracy (Fig. 41.4). To make sure that accuracy is not compromised, scaling series combines annealing with iterative refinements as we detail below.

The Static Case Algorithm. Let us first consider the case, where the prior $\overline{bel}$ is uniform. Scaling series starts by uniformly populating the space with very broad particles (i. e., with large δ). It evaluates the annealed measurement model for each particle and prunes out low probability regions of space. Then, it refines the estimate with slightly narrower particles, and repeats again and again until a sufficiently accurate estimate is obtained. Peak width is controlled during iterations using annealing, to make sure that each particle is able to represent its δ-neighborhood well. As the value of δ decreases during iterations, the density of particles per volume of δ-neighborhood is kept constant to maintain good particle density near peaks. We can think of δ as peak width, which decreases (due to annealing) with iterations. At each iteration, the δ-neighborhood of each peak will have the same fixed number of particles.

The formal algorithm listing is given in Algorithm 41.1, where S_δ denotes a δ-neighborhood and $\dim X$ is the dimensionality of the state space. The algorithm relies on three subroutines. `Even_Density_Cover` samples a fixed number

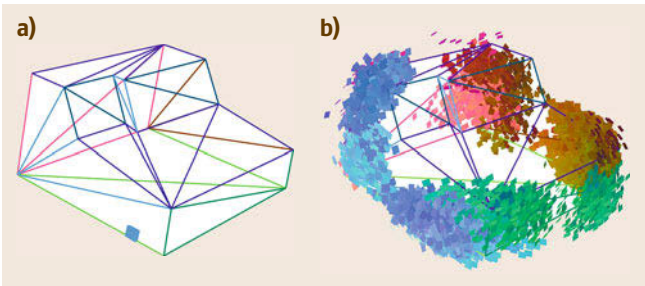


Fig.41.4a,b True (a) and annealed (b) belief for localization of the cash register [41.28]. The cash register model is shown as a wire frame. The small *colored squares* represent high likelihood particles. Note that annealing makes the problem much more ambiguous

of particles from each δ-neighborhood of a volume of space. It can be easily implemented using rejection sampling. `Importance_Weights` computes normalized importance weights using the annealed measurement model $p(\mathcal{D}|X)^{1/\tau}$. The `Prune` subroutine prunes out low probability regions. This step can be done either by weighted resampling or by thresholding on the weights.

The algorithm returns an approximation of the belief represented by a weighted particle set $\mathcal{X}$, where the weights $\mathcal{W}$ are set according to the measurement model, which in the case of uniform prior is proportional to the belief (41.8). Extension to the case of nonuniform prior can be done by simply multiplying the resulting weights by the prior. Sampling from the prior at the start of scaling series may also be useful. See VIDEO 721 for an example, where scaling series is successfully applied to 6-DOF localization of a cash register.

Algorithm 41.1 Scaling series algorithm for belief estimation

Input:
V_0 – initial uncertainty region,
$\mathcal{D}$ – data set,
M – number of particles per δ-neighborhood,
δ_* – terminal value of δ.

$\delta_0 \leftarrow Radius(V_0)$
$zoom \leftarrow 2^{-1/\dim X}$
$N \leftarrow \lfloor \log_2(Volume(S_{\delta_0})/Volume(S_{\delta_*})) \rfloor$
for $n = 1$ to N **do**
  $\delta_n \leftarrow zoom \cdot \delta_{n-1}$
  $\tau_n \leftarrow (\delta_n/\delta_*)^2$
  $\overline{\mathcal{X}}_n \leftarrow$ `Even_Density_Cover` (V_{n-1}, M)
  $\mathcal{W}_n \leftarrow$ `Importance_Weights` $(\overline{\mathcal{X}}_n, \tau_n, \mathcal{D})$
  $\mathcal{X}_n \leftarrow$ `Prune` $(\overline{\mathcal{X}}_n, \mathcal{W}_n)$
  $V_n \leftarrow$ `Union_Delta_Neighborhoods` $(\mathcal{X}_n, \delta_n)$
end for
$\mathcal{X} \leftarrow$ `Even_Density_Cover` (V_N, M)
$\mathcal{W} \leftarrow$ `Importance_Weights`$(\mathcal{X}, 1, \mathcal{D})$

Output:
$(\mathcal{X}, \mathcal{W})$ – a weighted particle set approximating the belief.

Dynamic Case. Scaling series can be extended to the dynamic case using the same technique as in the histogram filter. During the measurement update, the measurement model is estimated using scaling series with a uniform prior. This produces a set of weighted particles $\mathcal{X}_t$. During the dynamics update, the importance weights are adjusted to capture the motion model via the Bayesian recursion equation (41.9). To do this, for each particle X_t in $\mathcal{X}_t$, the importance weight is multiplied by the prior $\overline{bel}_t(X_t)$. Like in the histogram filter, the prior at a point X_t is computed by replacing the integral in (41.10) with a summation, except now the summation is done over all particles in $\mathcal{X}_{t-1}$. For other versions of dynamic scaling series, refer to [41.28, Chapter 2].

Other Advanced Inference Methods

Other advanced methods have been used to solve the tactile object localization problem, including the annealed particle filter (APF) [41.25], the GRAB algorithm [41.31], and the manifold particle filter (MPF) [41.32].

The APF algorithm is similar to scaling series as it also uses particles and iterative annealing. This algorithm was originally developed for articulated object tracking based on vision data [41.33, 34]. Unlike scaling series, APF keeps the number of particles constant at each iteration and the annealing schedule is derived from the particle set itself based on survival rate. Due to these properties, APF handles poorly in multimodal scenarios [41.35], which are prevalent in tactile object localization.

The GRAB algorithm is based on grids and measurement model bounds. It performs iterative grid refinements and prunes low-probability grid cells based on measurement model bounds. Unlike the majority of inference methods, GRAB is able to provide guaranteed results as long as the measurement model bounds are sound. It is well-suited for problems with many discontinuities, for example, whenever negative information is used in the measurement model. However, for smooth measurement models (such as the proximity model), GRAB has been shown to be slower than scaling series [41.28].

Similarly to the dynamic version of scaling series, the MPF algorithm samples particles from the measurement model and weighs them by the dynamics model. However, to sample from the measurement model, MPF draws a set of particles from the contact manifold, defined as the set of object states that contact a sensor without penetrating. Then, MPF weights these new particles by propagating prior particles forward using the dynamics model and applying kernel density estimation. Since the contact manifold is a lower dimensional manifold than the full state space of object poses, the MPF requires fewer particles than a traditional PF. It is not yet known if the improvement is sufficient to handle the full 6 DOF problem. However, it has been shown that MPF is capable of handling accurate sensors [41.32]. While MPF provides a method for sampling from the contact manifold for a single contact, it is unclear how to ex-

tend this approach to the case of multiple simultaneous contacts.

41.2.5 Planning

In order to localize an object, we need a strategy for gathering the data. Since each touch action can take significant time to execute, it is particularly important to accomplish the task with as few touches as possible. In this section, we will cover how to generate candidate actions and how to select which action to execute, with the goal of either localizing the object or grasping it with a high probability of task success.

Candidate Action Generation

The first step in selecting actions for tactile localization is to generate a set of candidate motions. While the object's pose is uncertain, we would like to avoid knocking over the object or damaging the robot. For this reason, actions are generally guarded motions, where the robot stops moving upon detecting contact. For multifingered robots, it may be advantageous to close the fingers upon detecting contact, in order to generate more than one contact point per action.

Candidate motions can be automatically generated in a similar manner to randomized or heuristic grasp planning methods (such as [41.36] or [41.37]). This approach is particularly useful if the goal is to generate multiple fingertip contacts, because any motion that is likely to grasp the object is also likely to result in informative contacts.

One option is to generate a small pool of good candidate motions based on the object shape, and to execute them relative to the current most-likely state of the object, as in [41.38]. Another option is to generate a larger pool of motions that are fixed with respect to the world, as in [41.39]. Finally, we can generate paths that specifically target disambiguating features. For example, *Schneiter* and *Sheridan* [41.40] show how to select paths that are guaranteed to disambiguate among remaining object shape and pose hypotheses, or if none such are possible, how to select paths that at least guarantee that something will be learned.

Bayesian Formulation of Planning

Once a pool of candidate actions has been created, the problem of selecting an optimal action to take next can be formulated as a partially observable Markov decision process (POMDP). Details on POMDPs can be found in Chap. 14. Briefly, a POMDP consists of a set of states $\mathcal{X} := \{X_i\}$, a set of actions $\mathcal{A} := \{A_j\}$, a set of observations $\mathcal{D} := \{D_k\}$. Both measurement model and dynamics model from Sect. 41.2.2 are now conditioned on the action A taken. This change is clearly visible

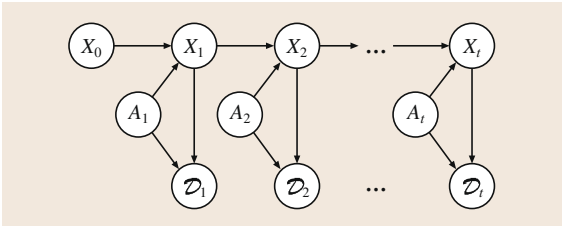


Fig. 41.5 Bayesian network representation of the planning problem. Note how it differs from Fig. 41.1b. Measurements $\mathcal{D}_t$ now depend on both the state X_t and the action chosen A_t. The state X_t evolves based on both the prior state X_{t-1} and the chosen action A_t

in the dynamic Bayesian network representation of the problem: compare Fig. 41.5 to Fig. 41.1b. Hence, the dynamics model is now $p(X_t|X_{t-1}, A_t)$, which is often called the transition model in the context of POMDPs. It represents the probability distribution of state X_t, given that we were in state X_{t-1} and executed action A_t. Similarly, the measurement model is now $p(D|X, A)$. For problems where the state X is continuous, the state space is typically discretized using either a grid or a set of particles.

For tactile object localization, the states are the actual object poses, the actions are our candidate robot motions (which may or may not touch the object), and the observations can include tactile and/or proprioceptive information gained while executing actions. We do not actually know the true underlying object pose, but at any given time t, we can estimate the uncertain belief bel_t about the object pose using methods described in Sect. 41.2.3. In the context of POMDPs, the belief is often referred to as the belief state.

A POMDP policy specifies an action A, for every possible belief. Intuitively, there is an optimal policy that would get the robot to the goal state most efficiently for any possible starting state. This optimal policy can be found using value or policy iteration algorithms. However, for the problem of tactile localization, these algorithms are typically computationally prohibitive for any practical discretization of the state space. Hence, this problem is usually solved using methods that approximate the optimal policy for a subset of states. For instance, online replanning using one-step or multistep lookahead is nonoptimal but usually provides good results.

For a subset of tactile localization problems, one-step lookahead has been shown to be nearly optimal [41.41]. Problems in this class satisfy two assumptions: (1) the object is stationary and does not move when touched (e.g., a heavy appliance), and (2) the actions being considered are a fixed set of motions regardless of the current belief. With these as-

sumptions, tactile localization can be formulated as an adaptive submodular information-gathering problem. Adaptive submodularity is a property of information-gathering problems where actions have diminishing returns, and taking additional actions never results in loss of information.

One-Step Lookahead

One way to select the next action is to consider the expected effect of each action on the belief about the object pose. Figure 41.6 shows examples of the effect of different actions on different beliefs.

Based on the possible object poses according to the current belief bel_t, we need to consider the possible outcomes of each action A. However, the same action can result in different outcomes depending on the actual object pose, which is not known to us exactly.

Simulation Experiments. Since we can not determine the effect of action A exactly, we approximate it by performing a series of simulation experiments. Let $\mathcal{X}_t$ be the current set of state hypotheses. For histogram filters, this can be the entire set $\mathcal{X}$. For particle filters, it is the current set of particles. For each particular object pose $X \in \mathcal{X}_t$, we can use a geometric simulation to determine where the hand will stop along a guarded motion A, and what observation D we are likely to see. Then, we can perform a Bayesian update of bel_t to obtain the simulated belief $\widetilde{bel}_{A,D}$ that would result if we chosen action A and observed D. Let $\mathcal{D}_A$ denote the set of all simulated observations we obtained for an action A using these experiments.

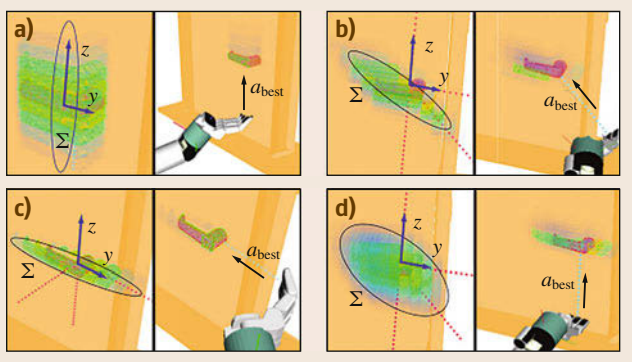


Fig.41.6a–d Selecting an action using a KLD-based utility function on examples of localizing a door handle. For each example, the starting belief bel_t is shown *on the left*. The action chosen $A := a_{\text{best}}$ and the resulting belief bel_{t+1} after executing the action are shown *on the right*. *Color* shows the probability of the state, ranging from high (*red*) to low (*blue*). **(a)** Sharp prior along the z-axis. **(b)** Sharp prior diagonal z–y-axis. **(c)** Sharp prior along the y-axis. **(d)** Wide prior along the diagonal z–y-axis (after [41.39])

To completely account for all possible outcomes, we need to simulate all possible noisy executions of action A and all possible resulting noisy observations D. However, restricting our consideration to the set of noise-free simulations works well in practice.

Each action A may also have an associated cost $C(A, D)$, which represents how expensive or risky it is to execute A. Note, that the action cost may depend on the observation D. For example, if the cost is the amount of time it takes to execute A, then depending on where the contact is sensed, the same guarded motion will stop at a different time along the trajectory.

Utility Function. In order to decide which action to take, we need a utility function $U(A)$ that allows us to compare different possible actions. The utility function should take into account both the usefulness of the action as well as its cost.

To localize the object, we want to reduce the uncertainty about its pose. Hence, we can measure action usefulness in terms of uncertainty of the resulting belief. Intuitively, the lower the uncertainty, the higher the usefulness of an action. The amount of uncertainty in a belief can be measured using entropy, which for a probability distribution $q(X)$ is defined as

$$H(q) := -\int q(X) \log q(X)\, \mathrm{d}X \,. \tag{41.11}$$

For each simulated belief $\widetilde{bel}_{A,D}$, we define its utility to be a linear combination of its usefulness (i. e., certainty) and cost

$$U(\widetilde{bel}_{A,D}) := -H(\widetilde{bel}_{A,D}) - \beta C(A, D) \,, \tag{41.12}$$

where β is a weight that trades off cost and usefulness [41.39]. Since we do not know what observation we will get after executing action A, we define its utility to be the expected utility of the resulting belief based on all possible observations

$$U(A) := \mathbb{E}_D\Big[U(\widetilde{bel}_{A,D})\Big] \,. \tag{41.13}$$

Based on the simulation experiments, this expectation can be estimated as a weighted sum over all the experiments

$$U(A) \approx \sum_{D \in \mathcal{D}_A} w_D U(\widetilde{bel}_{A,D}) \,, \tag{41.14}$$

where the weight w_D is the probability of the state X that was used to generate the observation D. More specifically, for a grid representation of state space, each $D \in \mathcal{D}_A$ was generated using some $X \in \mathcal{X}_t$, and thus, w_D

is the belief estimated for the grid point X. If $\mathcal{X}_t$ is represented by a set of weighted particles, then w_D is the weight of particle $X \in \mathcal{X}_t$ used to generate the observation D.

With this utility function in mind, the best action to execute is the one with the highest utility

$$A^{\text{best}} := \underset{A}{\text{argmax}}\Big[U(A)\Big] . \tag{41.15}$$

Since the utility function depends on negative entropy, methods that maximize this utility function are often called entropy minimization methods.

Alternative Utility Measures. Other measures of usefulness are also possible, such as reduction in average variance of the belief or Kullback–Leibler divergence (KLD) of the simulated belief from the current belief. However, KLD is only applicable for problems where the object is stationary as this metric does not support a changing state. An example of action selection using KLD metric is shown in Fig. 41.6.

If the goal is to successfully grasp the object, not just to localize it, then it makes more sense to select actions that maximize probability of grasp success rather than simply minimizing entropy. Success criteria for grasping can, for instance, be expressed geometrically as ranges in particular dimensions of object pose uncertainty that are required for a task to succeed. For a given belief, we can add up the probabilities corresponding to the object poses (states) for which executing a desired, task-directed grasp would succeed open-loop, and maximize that value rather than minimizing entropy. Action selection can still be done using one-step-lookahead. Using this metric instead of the one in (41.12) simply allows us to concentrate on reducing uncertainty along important dimensions, while paying less attention to unimportant dimensions [41.42]. For example, the axial rotation of a nearly cylindrical object is unimportant if it only needs to be grasped successfully.

Multi-step Planning in Belief Space

While in some situations one-step-lookahead methods give good results, in other situations it may be advantageous to look more than one step ahead. This is especially true if actions take a long time to execute, and/or if the plan can be stored and reused for multiple manipulation attempts.

Multi-step lookahead can be performed by constructing a finite-depth search tree (Fig. 41.7 for illustration). Starting with the current belief bel_t, we consider possible first-step actions $A_1, \ldots, A_3$ and simulate observations $D_{11}, \ldots, D_{32}$. For each simulated observation, we perform a belief update to obtain the simulated beliefs $B_{11}, \ldots, B_{32}$. For each of these simulated beliefs, we consider possible second-step actions (of which only $A_4, \ldots, A_6$ are shown). For each of the second-step actions, we again simulate observations and compute simulated beliefs. This process is repeated for third-step actions, fourth-step actions, and so on, until the desired number of steps is reached. The example in Fig. 41.7 shows the search tree for two-step lookahead.

After the tree is constructed from the top down to the desired depth, scoring is performed from the bottom up. Leaf beliefs are evaluated using the chosen utility metric. In the example in Fig. 41.7, the leafs shown are $B_{41}, \ldots, B_{62}$. For each leaf, we show the entropy H and cost βC. The utility of each corresponding action $A_4, \ldots, A_6$ is then computed by taking expectation over all observations that were simulated for that action. Note that we have to take expectations at this level because we have no control over which of the simulated observations we will actually observe. However, at the next level, we do have control over which action to execute, and we select the action with the maximum utility [41.42]. We compute the utility of beliefs $B_{11}, \ldots, B_{32}$ by subtracting their cost from the utility of the best action among their children. For example,

$$U(B_{22}) = U(A_6) - \beta C(A_2, D_{22}) . \tag{41.16}$$

In this manner, expectation and maximization can be carried out for multiple levels of the tree until the top is reached. Maximization at the very top level selects the most optimum action to execute next. Bold lines in Fig. 41.7 show the possible action sequences this algorithm would consider optimal depending on the possible observations made.

When searching to a depth greater than 1, branching on every possible observation is likely to be prohibitive. Instead, we can cluster observations based on how similar they are into a small number of canonical observations and only branch on these canonical observations.

Performing a multistep search allows us to include any desired final grasps as actions. These actions can result in early termination if the success criterion is met, but can also function as possible information-gathering actions if the criterion is not met. It also allows us to reason about actions that may be required to make desired grasps possible, such as reorienting the object to bring it within reach. As shown in [41.42], a two-step lookahead search is usually sufficient. Searching to a depth of 3 generally yields little additional benefit. One full sequence of information-gathering, grasping, and object reorientation using a multistep search with a depth of 2

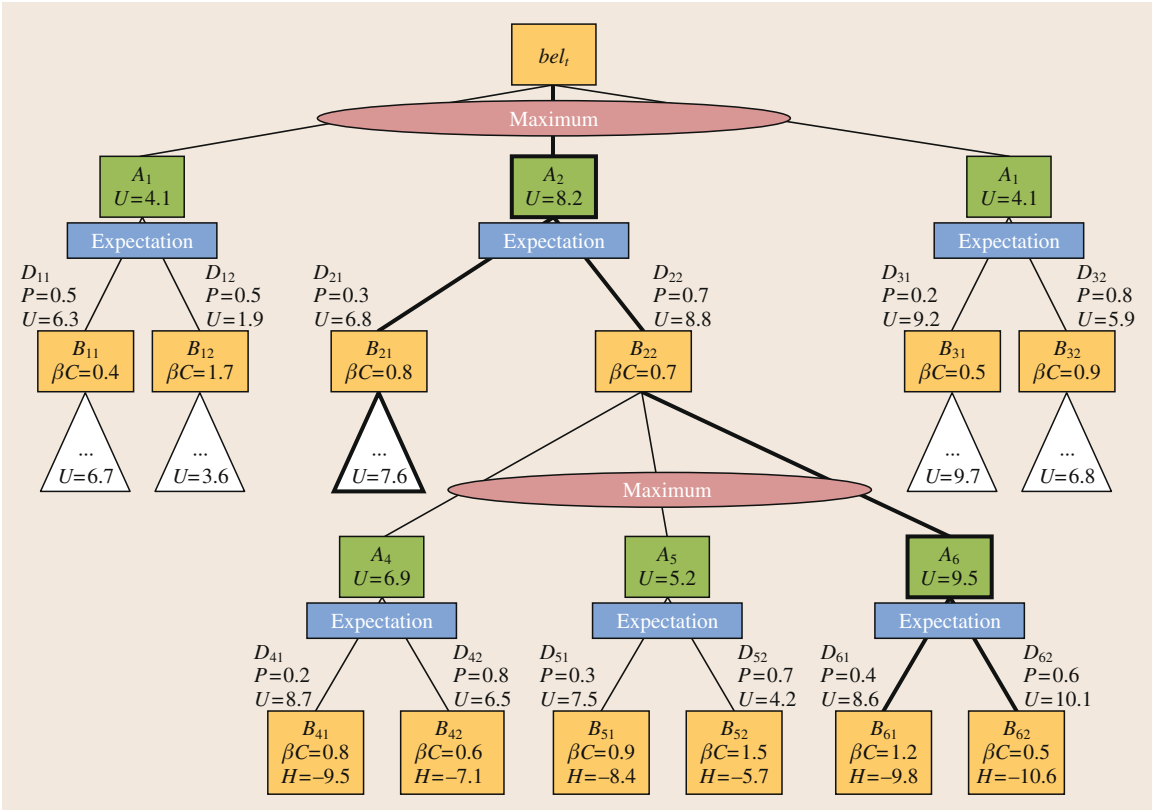


Fig. 41.7 Part of a depth-2 search tree through belief space, for three actions at each level and two canonical observations for each action; bel_t is the current belief. $A_1, \ldots, A_6$ are actions. $D_{11}, \ldots, D_{62}$ are simulated observations. And $B_{11}, \ldots, B_{62}$ are simulated beliefs. We can compute the entropy H and action cost $\beta\, C$ at the leafs, and thus compute the utility metric U for each leaf. Then, working upward from the bottom, we can compute the expected utility U of actions $A_4, \ldots, A_6$ by taking a weighted sum of their children. We then select the action with maximum utility (A_6 in this example) and use its utility $U(A_6)$ to compute the utility at node B_{22} using (41.16). The same operation is repeated for the upper levels, and at the top level we finally select the action with the highest utility as the best action to be performed (A_2 in this example) (after [41.42])

for grasping a power drill is shown in Fig. 41.8, and in VIDEO 77.

Other Planning Methods

There are many other possible methods for selecting actions while performing Bayesian state estimation for tactile object localization and grasping. As long as we continue to track the belief with one of the inference methods described above, even random actions can make progress toward successfully localizing the object. In this section, we will describe a few significantly different approaches to action selection for Bayesian object localization.

The planning methods we have described so far used either guarded motions or move-until-contact, to avoid disturbing or knocking over the object. However, one can instead use the dynamics of object pushing to both localize and constrain the object's pose in the process of push-grasping, as in *Dogar* and *Srinivasa* [41.43].

Also, instead of using entire trajectories as actions, we could consider smaller motions as actions out of which larger trajectories could be built. In this case, one- or two-step lookahead will be insufficient to get good results. Instead, we would need to construct plans consisting of many steps. In other words, we would need a much longer planning horizon. The challenge is that searching through belief space as described above is exponential in the planning horizon. In *Platt* et al. [41.44], the authors get around this problem by assuming the current most likely state is true and searching for longer horizon plans that will both reach the goal and also differentiate the current most likely hypothesis from other sampled, competing states. If we

Avidbots Ex1016
Page 1079 of 2154

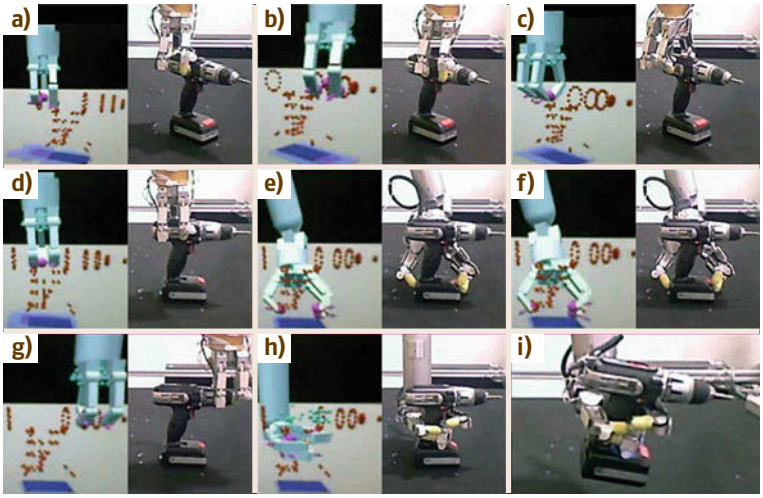


Fig.41.8a–i An example sequence of grasping a powerdrill based on a depth-2 search through belief space. The *right-hand side* of each pair of images shows the grasp just executed, while the *left-hand side* shows the resulting belief. The *dark blue flat rectangle* and the *red point-outline* of the powerdrill show the most likely state, while the *light blue rectangles* show states that are 1 standard deviation away from the mean in each dimension (x, y, θ). Panels **(a–c)** and **(e–g)** show information-gathering grasps being used to localize the drill, panel **(d)** shows the robot reorienting the drill to bring the desired grasp within reach, and panels **(h)** and **(i)** show the robot using the desired final grasp on the drill, lifting, and pulling the trigger successfully (after [41.42])

monitor the belief during execution and replan whenever it diverges by more than a set threshold, the goal will eventually be reached because each hypothesis will be either confirmed (and thus the goal reached) or disproved.

Finally, while the above methods can be used for planning for what we termed *global localization* (i. e., object localization under high uncertainty), if the object needs to be localized in an even broader context, as when trying to locate an object somewhere in a kitchen, then we may have to plan to gather information with higher level actions (e.g., opening cabinets). Such actions could be reasoned about using symbolic task planning, as in *Kaelbling* and *Lozano-Perez* [41.45].

41.3 Learning About an Object

While the previous section focused on localization of known objects, in this section, we discuss methods for learning about a previously unknown object or environment. The goal is to build a representation of the object that can later be useful for localization, grasp planning, or manipulation of the object. A number of properties may need to be estimated, including shape, inertia, friction, contact state, and others.

41.3.1 Shape of Rigid Objects

In this set of problems, the goal is to construct a 2-D or a 3-D model of the object's geometric shape by touching the object. Due to low data acquisition rate, touch-based shape reconstruction faces special challenges as compared to methods based on dense 3-D cameras or scanners. These challenges dictate the choice of object representation and exploration strategy.

While most of the work on shape reconstruction focuses on poking motions with the end-effector, some methods rely on techniques such as rolling the object in between planar palms equipped with tactile sensors. In these methods, shape of the object can be reconstructed from the contact curves traced out by the object on the palms. These methods were first developed for planar 2-D objects [41.46], and later extended to 3-D [41.47].

Representation

The shape reconstruction process is heavily dependent on the chosen representation. Since the data are sparse, a simpler shape representation can significantly speed up the shape acquisition process.

Shape Primitives. The simplest representation is a primitive shape: plane, sphere, cylinder, torus, tetrahe-

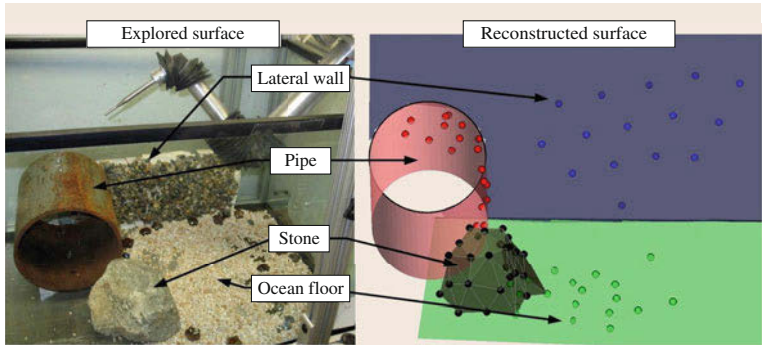


Fig. 41.9 A combination of primitive shapes and polygonal mesh is used for modeling objects during an underwater mapping experiment (after [41.48])

dra, or cube. In this case, only a few parameters need to be estimated based on the gathered data. For example, *Slaets* et al. estimate the size of a cube by performing compliant motions with the cube pressed against a table [41.49].

Patchwork of Primitives. If one primitive does not describe the whole object well, a *patchwork* of several primitives may be used. Parts of the object are represented with subsurfaces of different primitives, which are fused together [41.48] (Fig. 41.9), VIDEO 76. In this case, a strategy for refining the fusion boundaries between primitives is needed and additional data may be gathered for this specific purpose.

Super-quadrics. A slightly more complex primitive shape is a super-quadric surface, which gives more flexibility than a simple primitive. Super-quadric surfaces are surfaces described in spherical coordinates by

$$S(\omega_1,\omega_2) := \begin{pmatrix} c_x \cos^{\epsilon_1}(\omega_1)\ \cos^{\epsilon_2}(\omega_2) \\ c_y \cos^{\epsilon_1}(\omega_1)\ \sin^{\epsilon_2}(\omega_2) \\ c_z \sin^{\epsilon_1}(\omega_1) \end{pmatrix} . \quad (41.17)$$

The parameters c_x, c_y, and c_z describe the extent of the super-quadric along the axes x, y, and z, respectively. By adjusting the exponents ϵ_1 and ϵ_2, we can vary the resulting shape anywhere from an ellipse to a box. For example, setting $\epsilon_1, \epsilon_2 \approx 0$ results in box shapes; $\epsilon_1 = 1, \epsilon_2 \approx 0$ results in cylindrical shapes; $\epsilon_1, \epsilon_2 = 1$ results in ellipses. Hence, by estimating $c_x, c_y, c_z, \epsilon_1$, and ϵ_2, we can model a rich variety of shapes [41.50].

Polygonal Mesh. When none of the parametric representations capture the object's shape well enough, a polygonal mesh can be used. A polygonal mesh (or poly-mesh) can represent arbitrary shapes as it consists of polygons (typically, triangles) linked together to form a surface (Fig. 41.2). The simplest method is to link up collected data points to create faces (i. e., the polygons) of the mesh. However, this method tends to under-estimate the overall size of the object because data points at object corners are rarely gathered. This effect is present even for dense 3-D sensors, but especially noticeable here due to sparsity of the data. A more accurate representation can be obtained by collecting several data points for each polygonal face and, then, intersecting the polygonal faces to obtain the corners of the mesh. However, this tends to be a more involved, manual process [41.28].

Point Cloud. Objects can also be represented as clouds of gathered data points. In this case, the accuracy of the representation directly depends on the density of the gathered data and, hence, these methods typically spend a lot of time collecting data [41.51, 52].

Splines. Two-dimensional objects can be represented by splines. For example, *Walker* and *Salisbury* used proximity sensors and a planar robot to map smooth shapes placed on a flat surface [41.53] (Fig. 41.10).

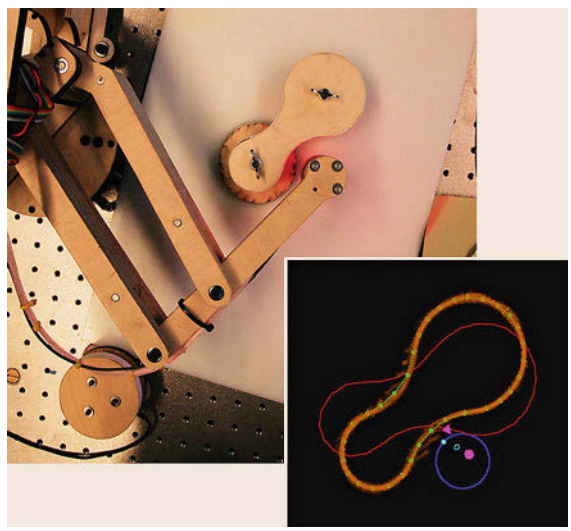


Fig. 41.10 A 2-D object is explored and represented by splines (after [41.53])

Volumetric Grid Maps. Objects can also be represented by volumetric grids. For 2-D objects, this representation is called an occupancy grid map. For 3-D objects, it is called a voxel grid map. Each grid cell represents a small part of the surface (or space) and records a binary value (e.g., whether the cell is occupied or not). This value is estimated based on gathered data using probabilistic methods similar to the methods used for mobile robot mapping with a 2-D laser [41.48, 54] (Fig. 41.11). For additional information on occupancy grid maps see Chap. 45.

Data Gathering

Different types of data can be collected during mapping. Tactile array images can be very informative for mapping planar objects. Sensed end-effector contact gives 3-D data points on the object surface. Link contacts can also be estimated and used for mapping even without sensory skin (Sect. 41.3.4 for more details). Moreover, the volume swept by the robot arm provides negative information, that is, the absence of obstacles in this volume. All of this information can be used both for mapping and for planning the next most informative move.

Guided Exploration. In many applications, it is acceptable to have a human operator guide the collection of data. This method allows for more accurate model building because humans can select the best points to sense on the object. Humans can also build variable accuracy models, where more data are collected near areas/features of interest and less data are collected elsewhere.

Autonomous Exploration. When autonomous model-building is required, a number of exploration strategies exist. The simplest approach is to select the sensing locations randomly. However, when using random sampling, the density of the gathered data can vary. An exhaustive strategy can ensure that points are gathered with a specified density. A triangular lattice arrangement has been shown to be the most optimal for such a strategy; however, this method takes considerable time. The most optimal decision about the next sensing location can be made by considering the amount of information that can be gained. These techniques are similar to the techniques described in Sect. 41.2.5. Several primitive and advanced exploration strategies for tactile mapping are compared in [41.48].

41.3.2 Articulated Objects

In all the prior sections, we were concerned with rigid objects, which tend to be simpler to model. The next step up in terms of complexity are objects comprised of several rigid parts, which can move with respect to each other. These objects are called articulated objects. For many objects, the motion of connected parts with respect to each other is restricted to only one or two DOFs by prismatic or revolute joints. Examples of articulated objects include simple objects such as doors, cabinet drawers, and scissors, as well as more complex objects such as robots and even humans (Fig. 41.12).

Manipulation can play an important role in modeling novel articulated objects. By applying a force to one part of an articulated object, we can observe how the other parts move and thus infer relationships between parts. Observation can be performed via the sense of touch or via some other sense, for example, vision. For example, if a robot arm pulls/pushes a door compliantly, its trajectory allows the robot to determine the width of the door [41.57]. Alternatively, if a robot arm pushes one handle of a pair of scissors, the location of the scissors joint can be inferred by tracking visual features with an overhead camera (Fig. 41.13).

A kinematic structure consisting of several revolute and prismatic joints can be represented as a relational model, which describes relationships between object parts. In order to build a relational model of a multijoint object, a robot needs an efficient exploration strategy for gathering the data. The exploration task can be described using a Markov decision process, similar to the ones discussed in Sect. 41.2.5. Since the goal is

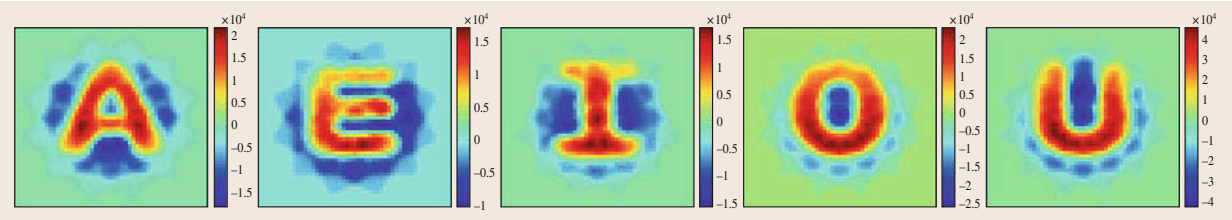


Fig. 41.11 Reconstruction of letters from tactile images. Pixels are colored by probability of occupancy in log-odds form. This probability can be seen to be high on the letters themselves, to be low in the surrounding area where measurements were taken, and to decay to the prior probability in the surrounding un-sensed area (after [41.54])



Fig. 41.12 Robot manipulating a variety of common articulated objects (*from left to right*): a cabinet door that opens to the right, a cabinet door that opens to the left, a dishwasher, a drawer, and a sliding cabinet door (after [41.55])

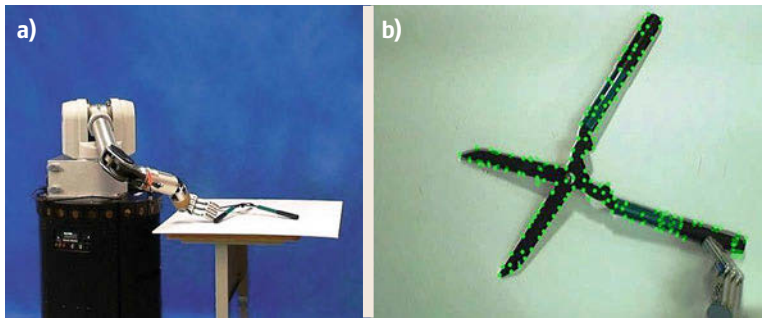


Fig.41.13a,b Robot exploring an articulated object (after [41.56]) **(a)** a robot arm pushes one handle of a pair of scissors to infer the location of its joint, **(b)** the view from the robot's camera with tracked visual features shown as green dots

to learn relational models, these Markov decision processes (MDPs) are called relational Markov decision processes (RMDPs). With the aid of RMDPs, robots can learn efficient strategies for exploring previously unknown articulated objects, learning their kinematic structure, and later manipulating them to achieve the goal state [41.56].

Some kinematic structures do not consist entirely of revolute and prismatic joints. A common example is garage doors, which slide in and out along a curved trajectory. General kinematic models can be represented by Gaussian processes, which can capture revolute, prismatic and other types of connections between parts [41.55], VIDEO 78.

41.3.3 Deformable Objects

Even more complex than articulated objects are deformable objects, which can be moved to form arbitrary shapes. The simplest of these are one-dimensional (1-D) deformable objects, such as ropes, strings, and cables. These objects are typically modeled as line objects with arbitrary deformation capability (except for stretching). A number of approaches can be found in [41.58].

A more complex type of deformable objects are planar deformable objects, such as fabric or deformable plastic sheets. While these objects can be represented as networks of nodes, modeling interaction between these nodes can get expensive and is not always necessary. For example, *Platt* et al. developed an approach to map planar deformable objects by swiping them between the robot's fingers [41.59] (Fig. 41.14). These maps can later be used for localization during subsequent swipes, in a manner similar to indoor robot localization on a map.

The most complex deformable objects are 3-D deformable objects. This category includes a great variety of objects in our everyday environments: sponges, sofas, and bread are just a few examples. Moreover, in medical robotics, tissues typically have to be modeled as 3-D deformable objects. Since the stiffness and composition of these objects can vary, they need to be modeled as networks of nodes, where different interaction can take place between distinct nodes. Such

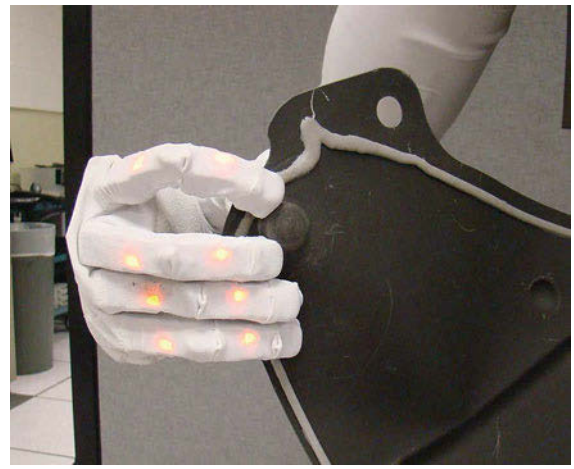


Fig. 41.14 Robot exploring a 2-D deformable object (after [41.59])

Avidbots Ex1016
Page 1083 of 2154

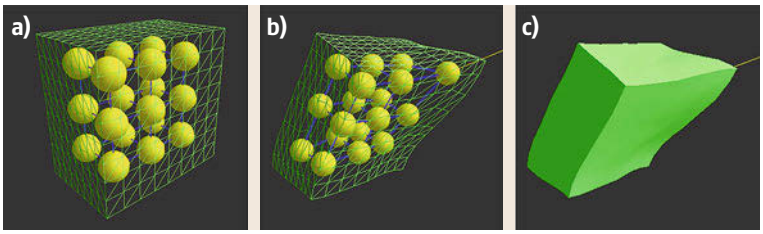


Fig.41.15a–c Model of a 3-D deformable object. The object at rest **(a)**. The object under influence of an external force **(b,c)** (after [41.60]

models typically require a lot of parameters to describe even reasonably small objects. Although, some efficiency improvements can be obtained by assuming that subregions of the object have similar properties. Despite the challenges, 3-D deformable object modeling has many useful applications, for example, a surgery simulator or 3-D graphics. *Burion* et al. modeled 3-D objects as a mass-spring system (Fig. 41.15), where mass nodes are interconnected by springs [41.60]. By applying forces to the object, we can observe the deformation and infer stiffness parameters of the springs (i. e., elongation, flexion, and torsion).

41.3.4 Contact Estimation

Up until now, we have assumed that the contact location during data gathering was somehow known. However, determining the contact point is not as straightforward for robots as it is for humans, because robots are not fully covered in sensory skin. Thus, during interaction between a robot and its environment, nonsensing parts of the robot can come in contact with the environment. If such contact goes undetected, the robot and/or the environment may be damaged. For this reason, touch sensing is typically performed by placing sensors at the end-effector and ensuring that the sensing trajectories never bring nonsensing parts of the robot in contact with the environment.

Clearly, end-effector-only sensing is a significant constraint, which is difficult to satisfy in real-world conditions. Work on whole-body sensory skin is ongoing [41.61], yet for now only small portions of the robot's surface tend to be covered with sensory skin.

In lieu of sensory skin, contact can be estimated from geometry of the robot and the environment via active sensing with compliant motions. Once a contact between the robot and the environment is detected (e.g., via deviation in joint torques or angles), the robot switches to the active sensing procedure. During this procedure, the robot performs compliant motions (either in hardware or software), by applying a small force toward the environment and moving back and forth. Intuitively, the robot's body carves out free space as it moves, and thus, via geometrical reasoning, the shape of the environment and contact points can be deduced. However, from a mathematical perspective, this can be done in several different ways.

One of the earliest approaches was developed by *Kaneko* and *Tanie*, who formulated the self-posture changability (SPC) method [41.63]. In SPC, the intersection of finger surfaces during the compliant motions is taken to be the point of contact. This method works well in areas of high curvature of the environment (e.g., corners), but can give noisy results in areas of shallow curvature. In contrast to SPC, the space sweeping algorithm by *Jentoft* and *Howe* marks all points on finger surfaces as possible contact, and then gradually rules out the possibilities [41.62] (Fig. 41.16). The space sweeping algorithm provides less noisy estimates in areas of shallow curvature, but can be more susceptible to even the smallest disturbances of the environment's surface.

Mathematically, contact estimation can also be formulated as a Bayesian estimation problem. If the robot moves using compliant motions as described above, the

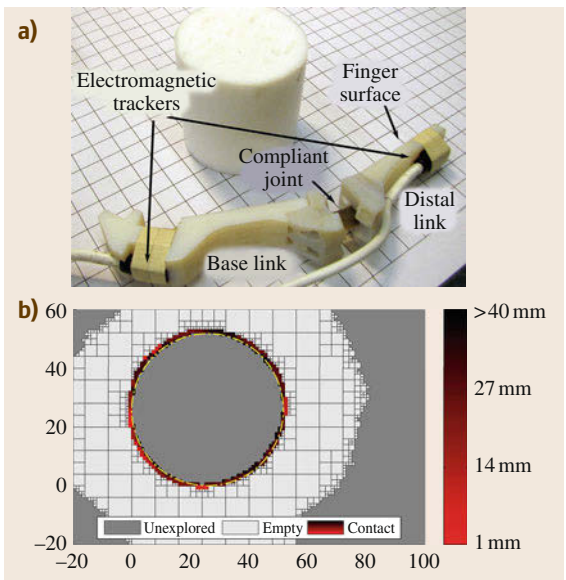


Fig. 41.16 **(a)** A two-link compliant finger used for contact sensing. **(b)** The resulting model built by the space sweeping algorithm (after [41.62])

sensor data are the joint angles of the robot. The state is the shape and position of the environment, represented using one of the rigid object models described in Sect. 41.3.1. Then, each possible state can be scored using a probabilistic measurement model, for example, the proximity model described in Sect. 41.2.2, and the problem can be solved using Bayesian inference methods [41.64].

41.3.5 Physical Properties

Up until now, we have focused on figuring out the shape and articulation of objects. While there are some advantages to doing so with tactile data as opposed to visual data, visual data can also be used to generate shape and articulation models. However, there are many properties of objects that are not easily detectable with vision alone, such as surface texture and friction, inertial properties, stiffness, or thermal conductivity.

Friction and Surface Texture

Surface friction, texture, and roughness can be estimated by dragging a fingertip over the object. The forces required to drag a fingertip over a surface at different velocities can be used to estimate surface friction parameters [41.65–67]. Accelerometer or pressure data is useful for estimating surface roughness (by the amplitude and frequency of vibration while sliding) [41.67, 68], or more directly identifying surface textures using classification techniques such as support vector machines (SVMs) or k-nearest neighbours (k-NNs) [41.69]. The position of the fingertip being dragged across the surface is also useful for estimating surface roughness and for mapping out small surface features [41.65]. Static friction is also often estimated as a by-product of trying to avoid object slip during a grasp, based on the normal force being applied at the moment when slip happens [41.70, 71]. Finally, the support friction distribution for an object sliding on a surface can be estimated by pushing the object at different locations and seeing how it moves [41.53, 72].

Inertia

Atkeson et al. [41.73] estimate the mass, center of mass, and moments of inertia of a grasped object, using measurements from a wrist force-torque sensor. Objects that are not grasped but that are sitting on a table can be pushed; if the pushing forces at the fingertips are known, the object's mass as well as the center of mass (COM) and inertial parameters in the plane can be estimated. *Yu* et al. [41.74] do so by pushing objects using two fingertips equipped with force/torque sensors and recording the fingertip forces, velocities, and accelerations. *Tanaka* and *Kushihama* [41.75] estimate the object's mass by pushing the object with known contact force and watching the resulting motion with a camera.

Stiffness

Identifying the stiffness (or the inverse, compliance) of an object is useful in terms of modeling the material properties of an object, but more directly, it is also useful for preventing the robot from crushing or damaging delicate objects. Gently squeezing an object and measuring the deflection seen for a given contact normal force is one method often used to estimate stiffness [41.68, 76, 77]. *Omata* et al. [41.78] use a piezoelectric transducer and a pressure sensor element in a feedback circuit that changes its resonant frequency upon touching an object, to estimate the object's acoustic impedance (which varies with both object stiffness and mass). *Burion* et al. [41.60] use particle filters to estimate the stiffness parameters of a mass-spring model for deformable objects, based on the displacements seen when applying forces to different locations on the object.

Thermal Conductivity

Thermal sensors are useful for identifying the material composition of objects based on their thermal conductivity. Fingertips with internal temperature sensors [41.68, 79] can estimate the thermal conductivity of the object by heating a robot fingertip to above room temperature, touching the fingertip to the object, and measuring the rate of heat loss.

41.4 Recognition

Another common goal of touch-based perception is to recognize objects. As with more typical, vision-based object recognition, the goal is to identify the object from a set of possible objects. Some methods assume known object shapes and try to figure out which object geometry matches the sensor data best. Other methods use feature-based pattern recognition to identify the object, often with a bag-of-features model that does not rely on geometric comparisons to overall object shape. Finally, there are methods that use object properties such as elasticity or texture to identify objects.

41.4.1 Object Shape Matching

Any of the methods for shape-based localization can also be used for recognition, by performing localiza-

tion on all potential objects. In terms of the equations used in Sect. 41.2, if we are considering a limited number of potential objects with known mesh models, we can simply expand our state space to include hypotheses over object identity and pose, instead of just pose. If we add an object identifier (ID) to each state in the state vector $\boldsymbol{X}$, and use the appropriate mesh model for that object ID for the measurement model for each state, all of the equations used in the previous sections still hold. The object ID for the combined object shape and pose that best matches the collected sensor data is then our top object hypothesis.

Nonprobabilistic methods for localization and object recognition often involve geometrically pruning objects from a set of possible object shapes/poses that do not match sensed data. For instance, *Gaston* and *Lozano-Perez* [41.1] and *Grimson* and *Lozano-Perez* [41.2] recognize and localize polyhedral objects by using the positions and normals of sensed contact points in a generate-and-test framework: generate feasible hypotheses about pairings between sensed contact points and object surfaces, and test them for consistency based on constraints over pairs of sensed points and their hypothesized face locations and normals. *Russell* [41.80] classify tactile array images as point, line, or area contacts, and use the time-sequence of contacts seen when rolling the object over the array to prune inconsistent object shapes.

Other methods recognize objects by forming a model of the object, then matching the new model with database objects. For instance, *Allen* and *Roberts* [41.81] build superquadric models of the object from contact data, then match the superquadric parameters against a database of object models. *Caselli* et al. [41.82] build polyhedral representations of objects (one based on contact points and normals that represents the space that the true object must fit inside, and one based on just contact points that the true object must envelop), and use the resulting representations to prune incompatible objects in a database of polyhedral objects.

41.4.2 Statistical Pattern Recognition

Many methods for tactile object recognition use techniques from statistical pattern recognition, summarizing sensed data with features and classifying the resulting feature vectors to determine the identity of the object. For instance, *Schöpfer* et al. [41.84] roll a tactile array over objects along a straight line, compute features such as image blob centroids and second or third moments over the resulting time series of tactile images, and use a decision tree to classify objects based on the resulting features. *Bhattacharjee* et al. [41.85] use k-nearest-neighbors on a set of features based on time series images from a forearm tactile array to identify whether an object is fixed or movable, and rigid or soft, in addition to recognizing the specific object.

Tactile array data is very similar to small image patches in visual data, and so techniques for object recognition using tactile arrays can look much like object recognition techniques that use visual data. For instance, *Pezzementi* et al. [41.86] and *Schneider* and *Sturm* [41.83] both use a Bag-of-Features model on fingertip tactile array data to recognize objects. The process for tactile data is almost the same as for visual images: for each training object (image), a set of tactile array images (2-D image patches) is obtained (Fig. 41.17), features (which are vectors of descriptive statistics) based on each tactile image are computed, and the features for all objects in the training set are clustered to form a vocabulary of canonical features (Fig. 41.18). Each object is represented as a histogram of how often each feature occurs. When looking at a new unknown object, the features computed from the tactile array images are similarly used to compute a his-

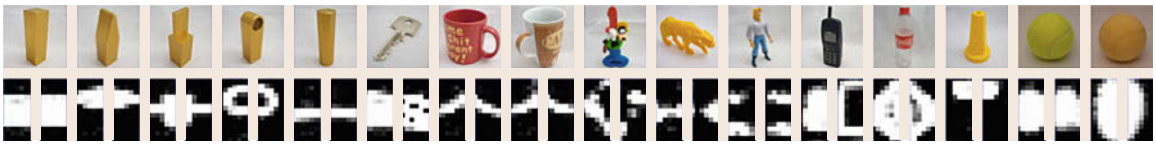


Fig. 41.17 Some objects and their associated tactile images for both left and right fingertips (after [41.83])

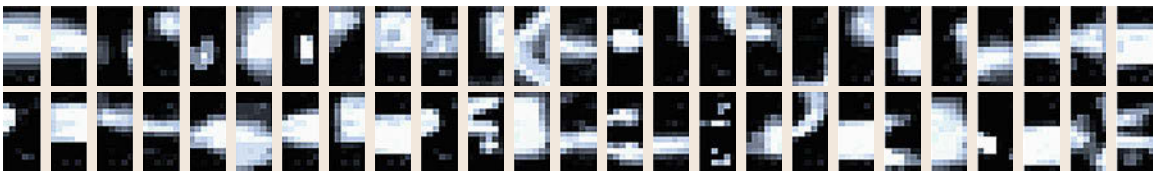


Fig. 41.18 Vocabulary of tactile images created using unsupervised clustering. Only the *left* finger's image is shown for each pair (after [41.83])

togram, and then the new histogram is compared against those of the objects in the database to find the best match. The one notable difference in the two domains is that tactile data is much more time-consuming and difficult to obtain than appropriate image patches: in 2-D visual object recognition, an interest point detection algorithm can quickly produce hundreds of salient patches from a given 2-D image, whereas for tactile object recognition, each tactile array requires touching the object, which in turn requires a planner (such as the ones described in Sect .41.2.5) to decide how to gather appropriate tactile data.

Tactile object exploration also allows one to gather and use information besides just tactile image patches: for instance, *Gorges* and *Navarro* [41.87] envelop objects in a multifingered hand, and use the resulting joint angles as features that directly relate to object dimensions, alongside additional features computed from tactile image patterns. In [41.88], *Pezzementi* and *Hager* combine techniques from vision-based object recognition with the type of Bayesian recognition-by-localization described in Sect. 41.4.1, using pairs of fingertip tactile image features along with their positions and surface normals to update a belief state over object poses and identities.

41.4.3 Recognition by Material Properties

Tactile object recognition sometimes has significant advantages over purely visual object recognition, particularly in cases where objects are visually similar but have different material properties detectable by tactile sensors. A number of recent methods classify objects using properties such as elasticity, surface texture, and thermal conductivity. Any of the methods listed in Sect. 41.3.5 can be used to identify objects, using many of the techniques described in Sect. 41.4.2. For instance, surface texture and roughness can be used to distinguish objects, as in [41.69] or [41.67], as can thermal conductivity, as in [41.79].

More specific properties can also be used to classify objects of particular types: for instance, *Chitta* et al. [41.89] identify bottles and also classify them as open or closed, full or empty by squeezing and rolling them from side to side and using decision tree classifiers on the resulting tactile measurements. *Drimus* et al. [41.90] also identify objects by squeezing them, but using k-nearest neighbor on the time series data of tactile measurements after performing dynamic time warping to align them.

41.4.4 Combining Visual and Tactile Sensing

Of course, visual object recognition also has advantages over purely tactile methods, and so combining data from both modalities can lead to even better object recognition and localization. For instance, *Allen* [41.91] uses sparse stereo data to provide regions of interest to explore with tactile sensors, then fuses the stereo data with the resulting tactile points to create surface and feature descriptions, which are then matched against objects in the database; object hypotheses are further verified with tactile exploration. *Boshra* and *Zhang* [41.92] recognize polyhedral objects by using tactile features and visual vertices and edges to formulate constraints in a constraint-satisfaction problem. *Nakamura* [41.93] grasps objects and learns object categories based on visual data (in the form of SIFT descriptors), audio information obtained while shaking (using a codebook of audio features based on mel-frequency cepstral coefficients), and haptic information obtained while squeezing (using hardness estimates based on finger pressure sensor readings); the learning and inference are done using probabilistic latent semantic analysis (PLSA).

Segmenting an object from a visual scene (which makes further efforts at recognition much easier) is also made easier by combining visual and tactile sensing; by pushing on an object and seeing which parts of the visual scene move together, objects can be segmented both from the background and from each other. For instance, *Gupta* and *Sukhatme* [41.94] use RGB-D data along with manipulation primitives such as spreading motions to segment and sort Duplo bricks, and *Hausman* et al. [41.95] segment textureless objects in RGB-D data by performing visual segmentation and then resolving ambiguities using pushing motions.

Work on categorizing objects by their affordances or functions often also uses both visual and tactile data. For instance, *Sutton* et al. [41.96] uses both shape-based reasoning (using shape derived from range images) as well as interaction-based reasoning (using data from the robot physically interacting with the object) to predict and confirm object affordances such as grasping, containing, and sitting (being suitable for humans to sit upon). *Bekiroglu* [41.97] performs inference on a Bayesian network using visual, proprioceptive, and tactile data to assess whether an object/grasp combination will be both stable and also good for a task such as pouring, hand-over, or dishwashing.

Avidbots Ex1016
Page 1087 of 2154

41.5 Conclusions

This chapter described perceptual methods that rely on manipulation. Although these methods are generally more complex than vision-based perception, they are useful in low-visibility conditions, in determining properties that require manipulation, and in situations where perception is needed for subsequent manipulation tasks. Perception-via-manipulation faces special challenges due to the low data acquisition rate and the fact that each sensing action disturbs the scene. To address these challenges, special inference methods have been developed to cope with situations where data are insufficient to fully constrain the problem (Sect. 41.2.3). Special planning methods had to be applied to minimize the number of sensing actions and reduce the possibility of significant scene disturbance or non-sensed contacts (Sect. 41.2.5). These challenges also dictate the choice of object models (Sect. 41.3), and play a role in object recognition methods (Sect. 41.4).

41.5.1 Real-Life Applications

Perception-*via*-manipulation is already used in a number of real-world applications and an even greater number of applications will likely benefit from it in the near future. In manufacturing, touch-based localization of parts for machining or assembly is often preferred to optical methods because it is not affected by optical properties of objects (e.g., transparency or reflectivity). At present, low uncertainty in part positioning is required prior to touch-based localization, which leads to high instrumentation and reconfiguration costs. However, the new localization methods presented in this chapter can cope with high uncertainty and thus enable flexible manufacturing lines capable of producing a variety of parts without reconfiguration. In the future, fully automated in-home manufacturing may even become possible.

Medical robotics is another mature real-world application, in which robotic contact with the environment (i. e., the patient) is not only unavoidable, it is in fact required for many of the procedures. Visibility is often poor, and sensing with devices that emit sound, light, or other waves can be harmful to the patient. Under these conditions, sensing-via-contact can sometimes be the most viable option. The environment is inherently deformable, and thus, medical robotics has been a strong driver for research on manipulation and perception of deformable objects.

Another area in which robots are indispensable is underwater robotics. It is not always safe or convenient for humans to be performing tasks deep underwater, and hence, remotely operated robots (ROVs) are a natural choice. ROVs even made international news during recovery operations from the BP oil spill disaster in the spring of 2010. Since visibility underwater is often poor, perception-via-contact plays an even more important role.

Beyond the more established robotic applications, as the fields of service robotics and disaster recovery develop, the necessity of sensing-via-contact becomes more and more obvious. In these applications, robots often have to work in clutter and in close proximity to humans. Under these conditions, visibility can be poor, yet at the same time, unsensed contact can be disastrous.

41.5.2 Future Directions of Research

Inspired by the many existing and potential applications, the field of perception-by-manipulation is evolving and receiving more attention. Despite the challenges, there are bountiful opportunities for further fruitful research:

- The vast majority of touch-based perception work deals with rigid objects, whereas many objects in our environments are articulated and/or deformable. These types of objects need to be studied in more detail.
- For rigid objects, touch-based perception of moving objects, object piles, and objects in clutter has been under-explored.
- Whole-body contact estimation needs further attention, with or without sensory skin. The challenges here are significant, but good solutions would increase the safety of robots in service applications and other unstructured environments.
- Last but not least, while Bayesian methods have found wide acceptance in other areas of robotics, touch-based perception has yet to fully benefit from these methods. Thus, application of Bayesian methods to any of the above listed areas is particularly promising.

41.5.3 Further Reading

For a broader view, a lot of related material can be found in other chapters throughout this handbook. Most sensing-via-manipulation is done using contact or near-contact (i. e., close-range) sensors. These sensors are described in Chap. 28. A few approaches use classical vision sensors in conjunction with manipulation (as in

Sect. 41.3.2, for example). Vision sensors and methods are covered in Chaps. 32 and 34. Manipulator design and control are covered in Chaps. 4, 8, and 9. Other relevant chapters include 7, 14 and 47 for planning methods, 37, 45, and 46 for modeling, 33 for recognition, 5 and 35 for inference.

Video-References

VIDEO 76 Tactile exploration and modeling using shape primitives
available from http://handbookofrobotics.org/view-chapter/41/videodetails/76

VIDEO 77 Tactile localization of a power drill
available from http://handbookofrobotics.org/view-chapter/41/videodetails/77

VIDEO 78 Modeling articulated objects using active manipulation
available from http://handbookofrobotics.org/view-chapter/41/videodetails/78

VIDEO 721 Touch-based door handle localization and manipulation
available from http://handbookofrobotics.org/view-chapter/41/videodetails/721

VIDEO 723 6-DOF object localization via touch
available from http://handbookofrobotics.org/view-chapter/41/videodetails/723

References

41.1 P.C. Gaston, T. Lozano-Perez: Tactile recognition and localization using object models: The case of polyhedra on a plane, IEEE Trans. Pattern Anal. Machine Intell. **6**(3), 257–266 (1984)

41.2 W.E.L. Grimson, T. Lozano-Perez: Model-based recognition and localization from sparse range or tactile data, Int. J. Robotics Res. **3**, 3–35 (1984)

41.3 O.D. Faugeras, M. Hebert: A 3-D recognition and positioning algorithm using geometrical matching between primitive surfaces, Proc. 8th Intl. Jt. Conf. Artif. Intell., Los Altos (1983) pp. 996–1002

41.4 S. Shekhar, O. Khatib, M. Shimojo: Sensor fusion and object localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1986) pp. 1623–1628

41.5 Y.-B. Jia, M. Erdmann: Pose from pushing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 165–171

41.6 K.Y. Goldberg: Orienting polygonal parts without sensors, Algorithmica **10**(2-4), 201–225 (1993)

41.7 M.A. Erdmann, M.T. Mason: An exploration of sensorless manipulation, IEEE J. Robotics Autom. **4**(4), 369–379 (1988)

41.8 H.T. Yau, C.H. Menq: An automated dimensional inspection environment for manufactured parts using coordinate measuring machines, Int. J. Prod. Res. **30**(7), 1517–1536 (1992)

41.9 K.T. Gunnarsson, F.B. Prinz: CAD model-based localization of parts in manufacturing, Computer **20**(8), 66–74 (1987)

41.10 K.T. Gunnarsson: Optimal Part Localization by Data Base Matching with Sparse and Dense Data, Ph.D. Thesis (Dept. Mech. Eng., Carnegie Mellon Univ. Pittsburgh 1987)

41.11 H.J. Pahk, W.J. Ahn: Precision inspection system for aircraft parts having very thin features based on CAD/CAI integration, Int. J. Adv. Manuf. Technol. **12**(6), 442–449 (1996)

41.12 M.W. Cho, T.I. Seo: Inspection planning strategy for the on-machine measurement process based on CAD/CAM/CAI integration, Int. J. Adv. Manuf. Technol. **19**(8), 607–617 (2002)

41.13 Z. Xiong: Workpiece Localization and Computer Aided Setup System, Ph.D. Thesis (Hong Kong Univ. Sci. Technol., Hong Kong 2002)

41.14 J. Hong, X. Tan: Method and apparatus for determining position and orientation of mechanical objects, US Patent US520 8763 A (1993)

41.15 B.K.P. Horn: Closed-form solution of absolute orientation using unit quaternions, J. Opt. Soc. Am. A **4**(4), 629–642 (1987)

41.16 C.H. Menq, H.T. Yau, G.Y. Lai: Automated precision measurement of surface profile in CAD-directed inspection, IEEE Trans. Robotics Autom. **8**(2), 268–278 (1992)

41.17 Y. Chu: Workpiece Localization: Theory, Algorithms and Implementation, Ph.D. Thesis (Hong Kong Univ. Sci. Technol., Hong Kong 1999)

41.18 Z. Xiong, M.Y. Wang, Z. Li: A near-optimal probing strategy for workpiece localization, IEEE Trans. Robotics **20**(4), 668–676 (2004)

41.19 Y. Huang, X. Qian: An efficient sensing localization algorithm for free-form surface digitization, J. Comput. Inform. Sci. Eng. **8**, 021008 (2008)

41.20 L.M. Zhu, H.G. Luo, H. Ding: Optimal design of measurement point layout for workpiece localization, J. Manuf. Sci. Eng. **131**, 011006 (2009)

41.21 L.M. Zhu, Z.H. Xiong, H. Ding, Y.L. Xiong: A distance function based approach for localization and profile error evaluation of complex surface, J. Manuf. Sci. Eng. **126**, 542–554 (2004)

41.22 Y. Sun, J. Xu, D. Guo, Z. Jia: A unified localization approach for machining allowance optimization of complex curved surfaces, Precis. Eng. **33**(4), 516–523 (2009)

41.23 K. Gadeyne, H. Bruyninckx: Markov techniques for object localization with force-controlled robots, 10th Int. Conf. Adv. Robotics (ICAR) (2001) pp. 91–96

41.24 S.R. Chhatpar, M.S. Branicky: Particle filtering for localization in robotic assemblies with position uncertainty, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

41.25 C. Corcoran, R. Platt: A measurement model for tracking hand-object state during dexterous manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

41.26 K. Hsiao, L. Kaelbling, T. Lozano-Perez: Task-driven tactile exploration, Robotics Sci. Syst. Conf. (2010)

41.27 A. Petrovskaya, O. Khatib: Global localization of objects via touch, IEEE Trans. Robotics **27**(3), 569–585 (2011)

41.28 A. Petrovskaya: Towards Dependable Robotic Perception, Ph.D. Thesis (Stanford Univ., Stanford 2011)

41.29 A. Petrovskaya, O. Khatib, S. Thrun, A.Y. Ng: Bayesian estimation for autonomous object manipulation based on tactile sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 707–714

41.30 S. Arulampalam, S. Maskell, N. Gordon, T. Clapp: A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking, IEEE Trans. Signal Process. **50**(2), 174–188 (2002)

41.31 A. Petrovskaya, S. Thrun, D. Koller, O. Khatib: Guaranteed inference for global state estimation in human environments, Mob. Manip. Workshop Robotics Sci. Syst. (RSS) (2010)

41.32 M.C. Koval, M.R. Dogar, N.S. Pollard, S.S. Srinivasa: Pose estimation for contact manipulation with manifold particle filters, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 4541–4548

41.33 J. Deutscher, A. Blake, I. Reid: Articulated body motion capture by annealed particle filtering, IEEE Conf. Comput. Vis. Pattern Recog. (CVPR) (2000)

41.34 J. Deutscher, I. Reid: Articulated body motion capture by stochastic search, Int. J. Comput. Vis. **61**(2), 185–205 (2005)

41.35 A.O. Balan, L. Sigal, M.J. Black: A quantitative evaluation of video-based 3D person tracking, 2nd Jt. IEEE Int. Workshop Vis. Surveill. Perform. Eval. Track. Surveill. (2005) pp. 349–356

41.36 K. Huebner: BADGr – A toolbox for box-based approximation, decomposition and grasping, Robotics Auton. Syst. **60**(3), 367–376 (2012)

41.37 A.T. Miller, P.K. Allen: Graspit! a versatile simulator for robotic grasping, IEEE Robotics Autom, Magaz. **11**(4), 110–122 (2004)

41.38 K. Hsiao, T. Lozano-Pérez, L.P. Kaelbling: Robust belief-based execution of manipulation programs, 8 Int. Workshop Algorithm. Found. Robotics (WAFR) (2008)

41.39 P. Hebert, T. Howard, N. Hudson, J. Ma, J.W. Burdick: The next best touch for model-based localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 99–106

41.40 J.L. Schneiter, T.B. Sheridan: An automated tactile sensing strategy for planar object recognition and localization, IEEE Trans. Pattern Anal. Mach. Intell. **12**(8), 775–786 (1990)

41.41 S. Javdani, M. Klingensmith, J.A. Bagnell, N.S. Pollard, S.S. Srinivasa: Efficient touch based localization through submodularity, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1828–1835

41.42 K. Hsiao, L.P. Kaelbling, T. Lozano-Pérez: Robust grasping under object pose uncertainty, Auton. Robots **31**(2/3), 253–268 (2011)

41.43 M. Dogar, S. Srinivasa: A framework for push-grasping in clutter. In: *Robotics: Science and Systems VII*, ed. by H.F. Durrant-Whyte, N. Roy, P. Abbeel (MIT, Cambridge 2011)

41.44 R. Platt, L. Kaelbling, T. Lozano-Perez, R. Tedrake: Efficient planning in non-Gaussian belief spaces and its application to robot grasping, Int. Symp. Robotics Res. (2011)

41.45 L.P. Kaelbling, T. Lozano-Pérez: Integrated task and motion planning in belief space, Int. J. Robotics Res. **32**(9/10), 1194–1227 (2013)

41.46 M. Erdmann: Shape recovery from passive locally dense tactile data, Proc. Workshop Algorithm. Found. Robotics (1998) pp. 119–132

41.47 M. Moll, M.A. Erdmann: Reconstructing the shape and motion of unknown objects with active tactile sensors, Algorithm. Found. Robotics, Vol. V (2003) pp. 293–309

41.48 F. Mazzini: Tactile Mapping of Harsh, Constrained Environments, with an Application to Oil Wells, Ph.D. Thesis (MIT, Cambridge 2011)

41.49 P. Slaets, J. Rutgeerts, K. Gadeyne, T. Lefebvre, H. Bruyninckx, J. De Schutter: Construction of a geometric 3-D model from sensor measurements collected during compliant motion, 9th. Proc. Int. Symp. Exp. Robotics (2004)

41.50 A. Bierbaum, K. Welke, D. Burger, T. Asfour, R. Dillmann: Haptic exploration for 3D shape reconstruction using five-finger hands, 7th IEEE-RAS Int. Conf. Hum. Robots (2007) pp. 616–621

41.51 S.R. Chhatpar, M.S. Branicky: Localization for robotic assemblies using probing and particle filtering, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2005) pp. 1379–1384

41.52 M. Meier, M. Schopfer, R. Haschke, H. Ritter: A probabilistic approach to tactile shape reconstruction, IEEE Trans. Robotics **27**(3), 630–635 (2011)

41.53 S. Walker, J.K. Salisbury: Pushing using learned manipulation maps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3808–3813

41.54 Z. Pezzementi, C. Reyda, G.D. Hager: Object mapping, recognition, and localization from tactile geometry, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5942–5948

41.55 J. Sturm, A. Jain, C. Stachniss, C.C. Kemp, W. Burgard: Operating articulated objects based on experience, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2739–2744

41.56 D. Katz, Y. Pyuro, O. Brock: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation, Robotics Sci. Syst. (2008)

41.57 C. Rhee, W. Chung, M. Kim, Y. Shim, H. Lee: Door opening control using the multi-fingered robotic hand for the indoor service robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

Avidbots Ex1016
Page 1090 of 2154

41.58 D. Henrich, H. Wörn: *Robot Manipulation of Deformable Objects* (Springer, Berlin, Heidelberg 2000)

41.59 R. Platt, F. Permenter, J. Pfeiffer: Using bayesian filtering to localize flexible materials during manipulation, IEEE Trans. Robotics **27**(3), 586–598 (2011)

41.60 S. Burion, F. Conti, A. Petrovskaya, C. Baur, O. Khatib: Identifying physical properties of deformable objects by using particle filters, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1112–1117

41.61 A. Jain, M.D. Killpack, A. Edsinger, C. Kemp: Reaching in clutter with whole-arm tactile sensing, Int. J. Robotics Res. **32**(4), 458–482 (2013)

41.62 L.P. Jentoft, R.D. Howe: Determining object geometry with compliance and simple sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3468–3473

41.63 M. Kaneko, K. Tanie: Contact point detection for grasping an unknown object using self-posture changeability, IEEE Trans. Robotics Autom. **10**(3), 355–367 (1994)

41.64 A. Petrovskaya, J. Park, O. Khatib: Probabilistic estimation of whole body contacts for multi-contact robot control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 568–573

41.65 A.M. Okamura, M.A. Costa, M.L. Turner, C. Richard, M.R. Cutkosky: Haptic surface exploration, Lec. Notes Control Inform. Sci. **250**, 423–432 (2000)

41.66 A. Bicchi, J.K. Salisbury, D.L. Brock: Experimental evaluation of friction characteristics with an articulated robotic hand, Lec. Notes Control Inform. Sci. **190**, 153–167 (1993)

41.67 J.A. Fishel, G.E. Loeb: Bayesian exploration for intelligent identification of textures, Frontiers Neurorobot. **6**, 4 (2012)

41.68 P. Dario, P. Ferrante, G. Giacalone, L. Livaldi, B. Allotta, G. Buttazzo, A.M. Sabatini: Planning and executing tactile exploratory procedures, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 1896–1903

41.69 J. Sinapov, V. Sukhoy: Vibrotactile recognition and categorization of surfaces by a humanoid robot, IEEE Trans. Robotics **27**(3), 488–497 (2011)

41.70 M.R. Tremblay, M.R. Cutkosky: Estimating friction using incipient slip sensing during a manipulation task, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 429–434

41.71 R. Bayrleithner, K. Komoriya: Static friction coefficient determination by force sensing and its application, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1639–1646

41.72 K.M. Lynch: Estimating the friction parameters of pushed objects, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1993) pp. 186–193

41.73 C.G. Atkeson, C.H. An, J.M. Hollerbach: Estimation of inertial parameters of manipulator loads and links, Int. J. Robotics Res. **5**(3), 101–119 (1986)

41.74 Y. Yu, T. Arima, S. Tsujio: Estimation of object inertia parameters on robot pushing operation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1657–1662

41.75 H.T. Tanaka, K. Kushihama: Haptic vision-vision-based haptic exploration, Proc. 16th Int. Conf. Pattern Recognit. (2002) pp. 852–855

41.76 H. Yussof, M. Ohka, J. Takata, Y. Nasu, M. Yamano: Low force control scheme for object hardness distinction in robot manipulation based on tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3443–3448

41.77 J. Romano, K. Hsiao, G. Niemeyer, S. Chitta, K.J. Kuchenbecker: Human-inspired robotic grasp control with tactile sensing, IEEE Trans. Robotics **27**, 1067–1079 (2011)

41.78 S. Omata, Y. Murayama, C.E. Constantinou: Real time robotic tactile sensor system for the determination of the physical properties of biomaterials, Sens. Actuators A **112**(2), 278–285 (2004)

41.79 C.H. Lin, T.W. Erickson, J.A. Fishel, N. Wettels, G.E. Loeb: Signal processing and fabrication of a biomimetic tactile sensor array with thermal, force and microvibration modalities, IEEE Int. Conf. Robotics Biomim. (ROBIO) (2009) pp. 129–134

41.80 R.A. Russell: Object recognition by a 'smart' tactile sensor, Proc. Aust. Conf. Robotics Autom. (2000) pp. 93–98

41.81 P.K. Allen, K.S. Roberts: Haptic object recognition using a multi-fingered dextrous hand, Proc. IEEE Conf. Robotics Autom. (ICRA) (1989) pp. 342–347

41.82 S. Caselli, C. Magnanini, F. Zanichelli, E. Caraffi: Efficient exploration and recognition of convex objects based on haptic perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1996) pp. 3508–3513

41.83 A. Schneider, J. Sturm: Object identification with tactile sensors using bag-of-features, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 243–248

41.84 M. Schöpfer, M. Pardowitz, R. Haschke, H. Ritter: Identifying relevant tactile features for object identification, Springer Tracts Adv, Robotics **76**, 417–430 (2012)

41.85 T. Bhattacharjee, J.M. Rehg, C.C. Kemp: Haptic classification and recognition of objects using a tactile sensing forearm, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4090–4097

41.86 Z. Pezzementi, E. Plaku, C. Reyda, G.D. Hager: Tactile-object recognition from appearance information, IEEE Trans. Robotics **27**(3), 473–487 (2011)

41.87 N. Gorges, S.E. Navarro: Haptic object recognition using passive joints and haptic key features, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2349–2355

41.88 Z. Pezzementi, G.D. Hager: Tactile object recognition and localization using spatially-varying appearance, Int. Symp. Robotics Res. (ISRR) (2011)

41.89 S. Chitta, J. Sturm, M. Piccoli, W. Burgard: Tactile sensing for mobile manipulation, IEEE Trans. Robotics **27**(3), 558–568 (2011)

41.90 A. Drimus, G. Kootstra, A. Bilberg, D. Kragic: Design of a flexible tactile sensor for classification of rigid and deformable objects, Robotics Auton. Syst. **62**(1), 3–15 (2012)

Avidbots Ex1016
Page 1091 of 2154

41.91 P. Allen: Integrating vision and touch for object recognition tasks, Int. J. Robotics Res. **7**(6), 15–33 (1988)
41.92 M. Boshra, H. Zhang: A constraint-satisfaction approach for 3D vision/touch-based object recognition, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (1995) pp. 368–373
41.93 T. Nakamura: Multimodal object categorization by a robot, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007) pp. 2415–2420
41.94 M. Gupta, G. Sukhatme: Using manipulation primitives for brick sorting in clutter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3883–3889
41.95 K. Hausman, F. Balint-Benczedi, D. Pangercic, Z.-C. Marton, R. Ueda, K. Okada, M. Beetz: Tracking-based interactive segmentation of textureless objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1122–1129
41.96 M. Sutton, L. Stark, K. Bowyer: Function from visual analysis and physical interaction: A methodology for recognition of generic classes of objects, Image Vis. Comput. **16**(11), 745–763 (1998)
41.97 Y. Bekiroglu: Learning to Assess Grasp Stability from Vision, Touch and Proprioception Ph.D. Thesis (KTH Roy. Inst. Technol., Stockholm 2012)

Avidbots Ex1016
Page 1092 of 2154

Multimedia Contents

42. Haptics

Blake Hannaford, Allison M. Okamura

The word *haptics*, believed to be derived from the Greek word *haptesthai*, means *related to the sense of touch*. In the psychology and neuroscience literature, haptics is the study of human touch sensing, specifically via *kinesthetic* (force/position) and *cutaneous* (tactile) receptors, associated with perception and manipulation. In the robotics and virtual reality literature, haptics is broadly defined as real and simulated touch interactions between robots, humans, and real, remote, or simulated environments, in various combinations. This chapter focuses on the use of specialized robotic devices and their corresponding control, known as *haptic interfaces*, that allow human operators to experience the sense of touch in remote (teleoperated) or simulated (virtual) environments.

42.1 **Overview** 1064
42.1.1 Human Haptics 1065
42.1.2 Application Examples 1066
42.2 **Haptic Device Design** 1068
42.2.1 Mechanisms 1068
42.2.2 Sensing 1069
42.2.3 Actuation and Transmission 1070
42.2.4 An Example Device 1070
42.3 **Haptic Rendering** 1071
42.3.1 Rendering Complex Environments 1072
42.3.2 Virtual Coupling 1073
42.4 **Control and Stability of Force Feedback Interfaces** 1073
42.5 **Other Types of Haptic Interfaces** 1075
42.5.1 Vibrotactile Feedback 1075
42.5.2 Contact Location, Slip, and Shear Display 1076
42.5.3 Slip and Shear 1076
42.5.4 Local Shape 1077
42.5.5 Surface Displays 1078
42.5.6 Temperature 1078
42.6 **Conclusions and Further Reading** 1079
References 1079

Haptic technology is intimately connected with robotics through its reliance on dexterous mechatronic devices and draws heavily on the theoretical foundations of manipulator design, actuation, sensing, and control. In this chapter, we begin with motivation for haptic interface design and use, including the basic design of a haptic interface, information about human haptics, and examples of haptic interface applications. Next, we review concepts in the mechatronic design of kinesthetic haptic interfaces, including sensors, actuators, and mechanisms. We then examine the control aspect of kinesthetic haptic interfaces, particularly the rendering of virtual environments and stable and accurate display of forces. We next review tactile displays, which vary widely in their design due to the many types of tactile information that can be presented to a human operator. Finally, we provide resources for further study of haptics.

Avidbots Ex1016
Page 1093 of 2154

42.1 Overview

Haptics is the science and technology of experiencing and creating touch sensations in human operators. Imagine trying to button a coat, shake someone's hand, or write a note without the sense of touch. These simple tasks become extremely difficult to perform without adequate haptic feedback. To improve human operator performance in simulated and teleoperated environments, haptic interfaces seek to generate a compelling sensation that the operator is directly touching a real environment.

Haptic interfaces attempt to replicate or enhance the touch experience of manipulating or perceiving a real environment through mechatronic devices and computer control. They consist of a haptic device (HD, a manipulandum with sensors and actuators) and a control computer with software relating human operator inputs to haptic information display. While the low-level design of haptic interfaces varies widely depending on the application, their operation generally follows the *haptic loop* shown in Fig. 42.1. First, the haptic device *senses* an operator input, which may be position (and its derivatives), force, muscle activity, etc. Second, the sensed input is applied to a virtual or teleoperated environment. For a virtual environment, the effect of the operator's input on virtual objects and the subsequent response to be displayed to the operator are computed based on models and a *haptic rendering* algorithm. In teleoperation, a manipulator that is remote in space, scale, or power attempts to track the operator's input. When the manipulator interacts with its real environment, haptic information to be relayed to the operators is recorded or estimated. Finally, *actuators* on the haptic device are used to physically convey touch sensations to the human operator. Based on the haptic feedback, whether through unconscious or conscious human control, or simply system dynamics, the operator input is modified. This begins another cycle of the haptic loop.

Despite the simplicity of the concept of haptic display, there exist many challenges to developing compelling haptic interfaces. Many of these are addressed through fundamental robotics theory and an understanding of human haptic capabilities. In general, haptic interface performance specifications are based on human sensing and motor control characteristics. One major challenge in artificially generating haptic sensations is that the human operator's motion should be unrestricted when there is no contact with a virtual or remote object. Haptic devices must allow the human operator to make desired motions, thus requiring back-drivability and sufficient degrees of freedom of motion. A variety of robotic designs are used in haptic devices, including exoskeletons, actuated grippers, parallel and serial manipulators, small-workspace *mouse*-like devices, and large-workspace devices that capture whole arm, and even whole body, movement. Another challenge is that humans integrate kinesthetic (force/position) and cutaneous (tactile) information with motion and control cues to form haptic perceptions. Haptic devices would ideally include both force and tactile displays, although this has been rarely done due to size and weight limitations of actuators. Because of the human's sensitivity to high-frequency information, for many haptic interfaces and applications, this loop must repeat at a high

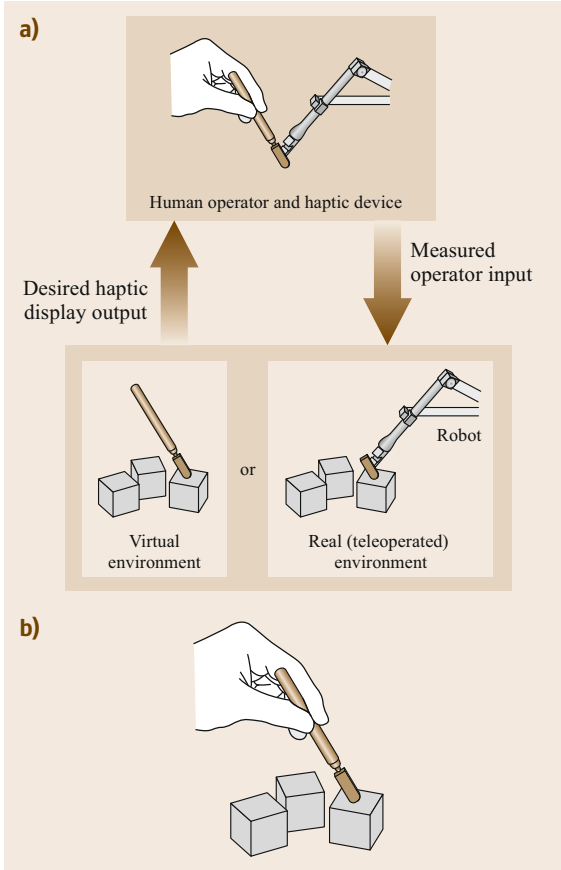


Fig. 42.1 (a) The *haptic loop* of a generic haptic interface. A haptic device senses human operator input, such as position or force, and the system applies this input to a virtual or teleoperated environment. The response of the environment to be relayed to the human operator is computed through models, haptic rendering, sensing, and/or estimation. Finally, actuators on the haptic device display corresponding touch sensations to the human operator. (b) The ideal result is that the human operator feels that he or she is interacting directly with a real environment

frequency, typically 1 kHz. Not only does a high update rate provide realistic (nondiscretized) touch sensations to the human operator, it also typically helps to maintain system stability. Controls analysis for haptic devices must consider both the continuous nature of the physical dynamics and the discrete nature of the computer control.

Before we examine the various components of haptic interfaces in detail, it is useful to motivate their design through a review of human haptics and applications of haptics. The remainder of this section is devoted to those topics.

42.1.1 Human Haptics

Two functions of the human nervous system play a primary role in haptics: kinesthesia (the internal sensing of forces and displacements inside muscles, tendons, and joints), and tactile sensing (the sensation of deformations of the skin).

Anatomy and Physiology

Haptics incorporates both, and is associated with an activity such as manipulation or exploration. Most of this chapter will address systems meant to interact primarily with the kinesthetic modality. Devices specifically aimed at tactile perception are described in Sect. 42.5. Even if tactile stimuli are not explicitly generated by a haptic device, tactile receptors are still stimulated, and are known to respond to frequencies as high as 10 000 Hz [42.1] and displacements as small as 2–4 μm [42.2–4].

Kinesthesia is mediated by muscle spindles, which transduce stretch of muscles, and Golgi tendon organs, which transduce joint rotation, especially at the extremes of motion. In principle, these and similar receptors could be stimulated directly to produce haptic sensations. For example, a vibration applied to a muscle tendon creates a strong sensation of muscle lengthening and corresponding joint motion in humans [42.5, 6]. Research in peripheral nerve stimulation for prosthesis control has demonstrated that electrodes implanted within individual fascicles of peripheral nerve stumps in amputees can be stimulated to produce sensations of touch or movement referred to the amputee's phantom hand [42.7].

Psychophysics

At the next level up from physiology and anatomy, psychophysics [42.8], the science of the physical capabilities of the senses, has been a rich source of design data for haptic device development. Its chief contribution has been methodologies that haptics researchers have applied to answer questions about what capabilities are needed in haptic devices. These sensory capabilities could then be translated into design requirements. Some of the chief psychophysical methods that have been fruitfully applied to haptics include threshold measurement by the method of limits and adaptive up-down methods. However, perception at threshold is not 100% reliable. Perception accuracy tends to depend on the strength of the stimulus, and the tradeoff between hit rate and false alarms depends strongly on the probability that a stimulus will be present in a given time interval (P_{stim}).

A more general notion is the receiver operating curve (Fig. 42.2, borrowed by psychophysics from radar theory), which plots the probability of a subject response given the existence of a stimulus, versus the probability of response given no stimulus. The curve is generated by measuring both probabilities at several different values of P_{stim}. The ideal response is the point (0,1): 100% response for stimuli and 0% response for nonstimuli. Human response is near this point for stimuli much above threshold, but declines to a rounded curve and eventually the 45° line as response below threshold becomes equal to chance.

Another relevant concept from psychophysics is the just noticeable difference (JND), commonly expressed as a percentage. This is the magnitude of a relative change in a stimulus, such as a force or displacement applied to the finger that is just perceivable by subjects.

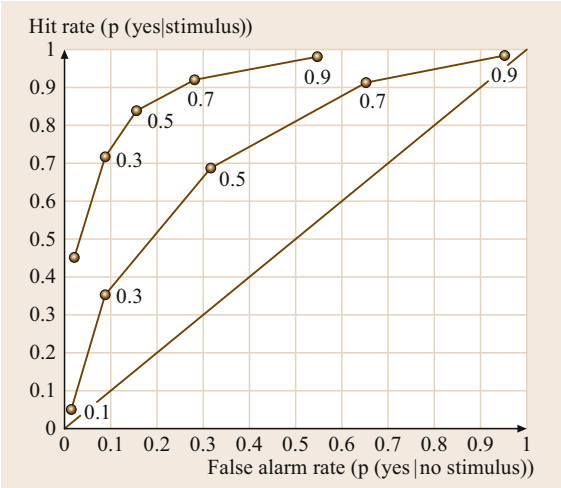


Fig. 42.2 Receiver operating curve (ROC). The ROC encodes the tradeoffs made by a subject between risk of false alarm and risk of missing a valid stimulus. Each point encodes a specific tradeoff between these two risks observed when stimuli are presented with a specified probability. ROCs for stronger signals tend towards the upper left hand corner (after [42.8] with permission from Lawrence Erlbaum and Associates, Mahwah)

Avidbots Ex1016
Page 1095 of 2154

For example, *Jones* [42.9] has measured JNDs of 6% for force applied to the human finger over a range of 0.5–200 N.

Psychology: Exploratory Procedures

In influential research starting in the 1980s, *Lederman* and *Klatzky* defined stereotyped hand motions called *exploratory procedures* (EPs), which are characteristic of human haptic exploration [42.10–12]. They placed objects into the hands of blindfolded subjects, and videotaped their hand motions. Their initial experiments [42.11] showed that the EPs used by subjects could be predicted based on the object property (texture, mass, temperature, etc.) that the subjects needed to discriminate. They also showed that the EPs chosen by subjects were the ones best able to discriminate that property. Furthermore, when asked to answer specific questions about objects (*Is this an eating utensil, further a fork?*), subjects used a two-stage sequence in which a more general lifting EP preceded more specific EPs [42.12].

Lederman and Klatzky's eight EPs (Fig. 42.3) and the property for which they are optimal are:

1. Lateral motion (texture)
2. Pressure (hardness)
3. Static contact (temperature)
4. Unsupported holding (weight)
5. Enclosure (global shape, volume)
6. Contour following (exact shape, volume)
7. Part motion test (part motion)
8. Function testing (specific function).

Each of these EPs is a bimanual task involving contact with all interior surfaces of the hand, motion of the wrist and various degrees of freedom of the hand, and tactile and temperature sensors in the skin (e.g., EPs 1 and 3), and kinesthetic sensors in the arm (EP 4). A haptic device capable of supporting all of these EPs is far beyond today's state of the art. However, the significance of these results for the design of haptic interface is great, since they allow us to derive device requirements from EPs.

42.1.2 Application Examples

The most common haptic device encountered by the general population is a vibration display device that provides haptic feedback while an operator plays a video game. For example, when the operator drives off the virtual road or bumps into a virtual wall, the hand controller shakes to imply driving over a rough surface or displays an impulse to represent the shock of hitting a hard surface. We examine two more pragmatic examples, medical simulators and computer-aided design (CAD) systems, in detail below. In addition, we review several commercially available haptic devices. Although haptic interfaces are not yet in widespread commercial use outside of entertainment, they are being integrated into numerous applications where the potential benefits are clear enough to justify the adoption of new technologies. A variety of novel and creative applications are being developed regularly in numerous fields, including:

- Assistive technology
- Automotive
- Design
- Education
- Entertainment
- Human–computer interaction
- Manufacturing/assembly
- Medical simulation
- Micro/nanotechnology
- Molecular biology
- Prosthetics
- Rehabilitation
- Scientific visualization
- Space
- Surgical robotics.

Medical Simulations

A major example driving much of today's haptic virtual environment research is simulation for training of hands-on medical procedures. Medical invasive thera-

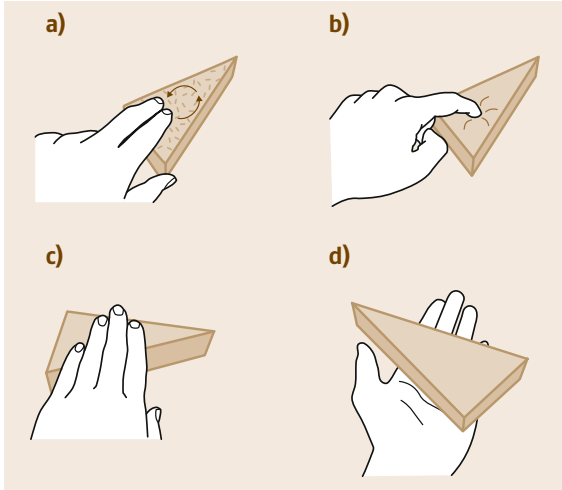


Fig.42.3a–d Four of the eight human exploratory procedures (EPs). **(a)** Lateral motion (texture); **(b)** pressure (hardness); **(c)** static contact (temperature); **(d)** unsupported holding (weight) (after *Lederman* and *Klatzky* [42.11])

Avidbots Ex1016
Page 1096 of 2154

peutic and diagnostic procedures, ranging from drawing blood samples to surgery, are potentially dangerous and painful for the patient and require the student to learn hands-on skills mediated by haptic information pathways [42.13]. Simulators both with and without haptic feedback aim to replace supervised learning directly on human patients or on animals. Simulators have proven highly effective in developing minimally invasive surgery skills [42.14], especially when haptic feedback is provided in early training [42.15]. Expected benefits of training with haptic simulators include:

- Reduced risk to patients both during and immediately after training
- Increased ability to simulate unusual conditions or medical emergencies.
- Ability to collect physical data during the training process and provide specific and directed feedback to the student.
- Increased training time per unit of instructor effort.

Approaches for simulator designs, specific medical applications, and training evaluation methods have also been widely studied in the last two decades, e.g., [42.16, 17]. However, the costs of this technology are still high. In addition, it is not always clear which improvements in simulator technology, such as haptic device performance or accuracy of soft-tissue modeling, lead to improved clinical performance and, ultimately, patient outcomes.

Computer-Aided Design

The Boeing Company [42.18] has studied the use of haptic interfaces for solving advanced problems in CAD. One such problem is verification of the ability to efficiently maintain a complex system such as an aircraft. In the past, mechanics could verify the procedures (such as change-out of parts) on a physical prototype. However, this analysis is difficult or impossible to perform visually on an advanced CAD system. The VoxMap Pointshell system (Fig. 42.4) was developed to allow test extraction of parts with a haptic interface. Force sensations from the haptic interface reproduce for the operator the physical constraints of the part bumping into elements of the complex workplace. If the operator can remove the part in the haptic interface, it is verified that this part can be maintained without undue disassembly of the aircraft. This capability has been proved useful in actual design activities.

Commercially Available Haptic Devices and Systems

There are a wide variety of haptic devices available from companies, although many researchers build their own haptic devices for special purposes. At the time of this writing, one of the most popular commercially available haptic devices is the Geomagic Touch [42.19], formerly known as the Phantom Omni from SensAble Technologies (Fig. 42.5). In the 1990s and 2000s, SensAble developed the Phantom line of stylus-type haptic devices. The Phantom Premium [42.20], a higher fidelity, larger workspace device, has also been used widely in haptics research. The high price of haptic devices (compared to visual displays) restricts the development of some commercial applications. The Geomagic Touch (Phantom Omni), which is an order of magnitude less expensive than the Phantom Premium, has gained popularity among haptics and robotics re-

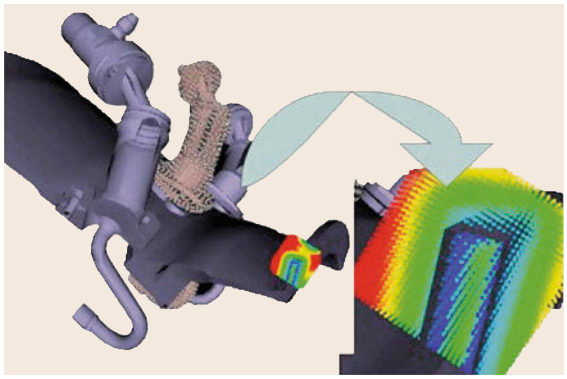


Fig. 42.4 Boeing computer-aided design (CAD) application for assembly and maintenance verification of complex aircraft systems. Boeing researchers developed the Voxmap/Pointshell software for haptically rendering very complex models in six degrees of freedom at high rates (courtesy of Bill McNeely, Boeing Phantom Works)



Fig. 42.5 The Phantom Omni device from SensAble Technologies, now marketed as the Geomagic Touch. This relatively low-cost device senses motion in six degrees of freedom from the stylus and can apply forces in the x, y, and z directions to the stylus tip (courtesy SensAble Technologies, Inc., Woburn)

Avidbots Ex1016
Page 1097 of 2154

searchers. In 2007, Novint Technologies [42.21] released the Novint Falcon, an inexpensive 3-DOF (three-degree-of-freedom) haptic device that is in turn an order of magnitude less expensive than the Phantom Omni.

Immersion has aimed at the mass market and consumer segments with a wide variety of haptics-based products, many of them involving a single degree of freedom. For example, they have licensed technology to makers of various video games, as well as mobile phone manufacturers, in the form of vibratory feedback in handheld devices and haptic-enabled steering wheels for driving games. Immersion also has a medical division selling medical simulators with haptic feedback.

Software for haptic rendering has also become widely available, through both commercial sources and research groups. Most companies that sell haptic devices also provide a standard development kit (SDK) with haptic rendering capability. In addition, not-for-profit open-source projects such as Chai3D [42.22] aim to make rendering algorithms from different groups publicly available, shortening application development time and allowing direct comparison of algorithms for benchmarking purposes.

42.2 Haptic Device Design

There are two broad classes of haptic devices: *admittance* and *impedance* devices. Admittance devices sense the force applied by the operator and constrain the operator's position to match the appropriate deflection of a simulated object or surface in a virtual world. In contrast, an impedance haptic device senses the position of the operator, and then applies a force vector to the operator according to computed behavior of the simulated object or surface.

Robots of the impedance type are back-drivable, have low friction and inertia, and have force-source actuators. A commonly used impedance haptic device in robotics-related research is the Phantom Premium [42.20, 23]. Robots of the admittance type, such as typical industrial robots, are non-back-drivable and have velocity-source actuators. The velocity is controlled with a high-bandwidth low-level controller, and is assumed to be independent of applied external forces. Some commercially available haptic devices, such as the HapticMaster [42.24], do operate under admittance control. While such closed-loop force control has been used for haptic display, more commonly designers have opted for mechanisms specially designed for open-loop force control to achieve simultaneously low cost and high bandwidth.

The choice of admittance or impedance architecture has many profound implications in the design of the software and hardware system. For a variety of reasons, including cost, the majority of haptic devices implemented today are of the impedance type. Because the preponderance of systems today are impedance devices and limitations on space, we limit our subsequent discussion to that class.

42.2.1 Mechanisms

Creating high-fidelity haptic sensations in the operator requires attention to mechanism design (Chap. 5). The requirements for impedance haptic devices are similar to those for designing manipulators suitable for force control. Desirable mechanism attributes for open-loop force control include low inertia, high stiffness, and good kinematic conditioning throughout a workspace designed to effectively match the appropriate human limb, primarily the finger or arm. The weight of the mechanism should be minimized, as it is perceived by the operator as weight and inertia of the virtual or teleoperated environment. Kinematic singularities (Chaps. 2, 5, and 18) are detrimental to haptic interfaces because they create directions in space in which the end-point cannot be moved by the human operator and thus impose disturbances on the illusion of haptic contact with virtual objects. High transmission ratios must be avoided as they introduce significant amounts of friction. This constraint requires haptic interfaces to make high demands on actuator performance.

Measures of Mechanism Performance

The ideal haptic device can move freely in any direction and is free of singular configurations as well as the *bad* effects of operating in their neighborhood. Traditionally, kinematic performance has been derived from the mechanism's Jacobian matrix, $\mathbf{J}(p, q)$, using some of the following well-known measures:

- Manipulability [42.25]: the product of the singular values of $\mathbf{J}(p, q)$
- Mechanism isotropy [42.26]: the ratio of smallest to the largest singular value of $\mathbf{J}(p, q)$
- Minimum force output [42.25, 27, 28]: maximizing the force output in the *worst* direction.

Dynamics can also be introduced into the cost function using measures such as dynamic manipulability [42.29]. This is still an active area of research and

there is no consensus yet on which dexterity measure is most appropriate for haptic devices.

Kinematic and Dynamic Optimization

This aspect of design requires synthesis of mechanisms that match the workspace of the (most often) human finger or arm while simultaneously avoiding kinematic singularities.

A haptic device workspace is defined to match that of the targeted human limb. This can be assisted by the use of anthropometric data [42.30]. The performance goals, such as low inertia and avoidance of kinematic singularities, must be formalized into a quantitative performance measure which can be computed for any candidate design. Such a measure must account for:

- Uniformity of kinematic conditioning throughout the target workspace
- Favoring designs with lower inertia
- Guaranteeing that the target workspace is reachable.

The measures of mechanism performance defined above operate at a single point in space and thus must be integrated over the entire workspace to derive a figure of merit for a proposed haptic device design. For example, if S is the set of all joint angles Θ, such that the end effector is inside the target workspace, one such measure is

$$M = \min_S W(\Theta) \,, \tag{42.1}$$

where $W(\Theta)$ is a measure of design performance. The performance measure should include a link length penalty such as

$$M = \min_S \frac{W(\Theta)}{l^3} \,, \tag{42.2}$$

in order to avoid solutions which have excessive size, compliance, and mass of long links. We could search a large family of mechanism designs to maximize M. For example, if a design has five free parameters (typically link lengths and offsets), and we study ten possible values for each parameter, 10^5 designs must be evaluated.

Available computing power has grown much faster than the complexity of realizable mechanisms on the human scale (as measured by their DOF). Thus, brute-force search of design spaces is often sufficient, and sophisticated optimization techniques are not necessary.

Grounded Versus Ungrounded Devices

Most current devices that provide kinesthetic feedback are physically grounded, that is, forces felt by the operator are with respect to the operator's *ground*, such as the floor or desktop. Ungrounded haptic feedback devices are more mobile and can operate over larger workspaces compared to grounded devices, which enables them to be used in large-scale virtual environments. A number of ungrounded kinesthetic feedback devices have been developed, for example [42.31–33]. Comparisons have been made between the performance of ungrounded and grounded haptic displays [42.34]. Some ungrounded devices provide tactile rather than kinesthetic sensations, and these are described in Sect. 42.5.

42.2.2 Sensing

Haptic devices require sensors to measure the state of the device. This state may be modified by the operator's applied position/force, the haptic control law, and/or device and environment dynamics. The operator's input is sensed in the form of an applied position or an applied force. Sensing requirements for haptics are similar to those of other robotic devices (Chap. 29) so only haptics-specific sensing issues are discussed here.

Encoders

Rotary optical quadrature encoders are typically used as position sensors on the joints of haptic devices. They are often integrated with rotary motors, which serve as actuators. The underlying sensing mechanism for encoders is described in Sect. 29.1. The required resolution of an encoder for a haptic device depends on the ratio between the angular distance of a single encoder tick to the end-point motion in Cartesian space. The resolution of the selected position encoder has effects beyond simple spatial resolution of the end-point, including the maximum stiffness that can be rendered (Sect. 42.4) without unstable or nonpassive behavior [42.35].

Many haptic applications, such as the rendering of virtual environments with damping (in which force is proportional to velocity), require velocity measurement. Velocity is typically obtained by numerical differentiation of the position signal obtained by an encoder. An algorithm for velocity estimation must be selected which is free of noise but minimizes phase lag at the frequencies of interest [42.36]. Thus, an alternative method is to use specialized hardware that measures the time between encoder ticks in order to compute the velocity [42.37].

Force Sensors

Force sensors are used in haptic devices as the operator input to an admittance-controlled device, or as a mechanism for canceling device friction and other undesirable dynamic properties in an impedance-controlled device.

Avidbots Ex1016
Page 1099 of 2154

When a force sensor such as a strain gauge or load cell measures the operator's applied force, care must be taken to thermally isolate the sensor, since thermal gradients in the sensor caused by body heat can affect force readings.

42.2.3 Actuation and Transmission

Haptic devices are differentiated from traditional computer input devices by actuators that are controlled to provide appropriate haptic sensations to the human operator. The performance of the haptic device depends heavily on the actuator properties and the mechanical transmission between the actuator and the haptic interaction point (HIP).

Requirements for Haptics

The primary requirements for actuators and mechanical transmission in impedance-type haptic devices are: low inertia, low friction, low torque ripple, backdriveability, and low backlash. In addition, if the design is such that the actuator itself moves as the user's position changes, a higher power-to-weight ratio is desired. Although closed-loop force control has been used for haptic display in impedance devices, most often the mechanism is designed to have sufficiently low friction and inertia so that open-loop force control is accurate enough.

One common mechanical transmission for haptic devices is the capstan drive (Fig. 42.6), which consists of smooth cables wrapped around pulleys of differing diameter to provide a gear ratio. A no-slip, high-friction contact between the cable and the pulleys is maintained through several wraps of the cable. The capstan drive minimizes friction forces felt by the human operator because it prevents translational forces on motor and joint axes.

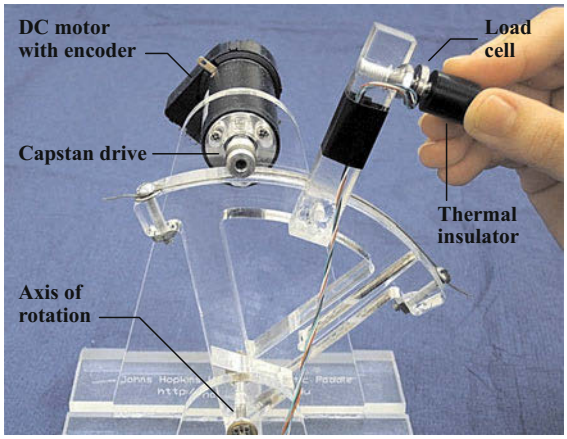


Fig. 42.6 This version of the Haptic Paddle [42.38] includes an encoder for position sensing, a single-axis load cell for force sensing, and a brushed motor with capstan transmission for actuation

Current amplifiers are typically used to create a direct relationship between the voltage output by the computer via a digital-to-analog (D/A) converter and the torque output by the motor. The effect of actuator and amplifier dynamics and D/A resolution on system stability is typically negligible in comparison to position sensor resolution and sampling rate for most haptic devices. Actuator or amplifier saturation can produce undesirable behavior, particularly in multi-DOF haptic devices where a single saturated motor torque may change the apparent geometry of virtual objects. The force vector, and thus the corresponding actuator torques, must be scaled appropriately if any actuator is saturated.

42.2.4 An Example Device

As an illustrative example, we will provide detailed design information for a simple 1-DOF haptic device known as the Haptic Paddle [42.38]. This section is meant to provide a concrete description of the types of components that are used in kinesthetic haptic devices, and the device can also be constructed following the instructions provided by Johns Hopkins University. Many widely available haptic devices share the common working principles of this device and differ chiefly in kinematic details arising from a greater number of DOF.

The haptic paddle shown in Fig. 42.6 is equipped with two sensors: a position a position encoder and a force sensor. A 500-counts-per-turn Hewlett-Packard HEDS 5540 encoder that is mounted directly on the motor. The quadrature process yields 2000 counts per revolution; the capstan transmission gear ratio and lever arm result in a position resolution of 2.24×10^{-5} m at the haptic interaction point (HIP). An optional load cell is used to measure the applied operator force. A plastic cap thermally insulates the load cell. In this device, the load cell can be used to minimize the effect of friction through a control law that attempts to zero the applied operator force when the HIP is not in contact with a virtual object.

The Haptic Paddle shown uses a brushed DC (direct current) motor with an aluminum pulley attached to the shaft. Like many commercial haptic devices, it uses a capstan drive: a cable is wrapped several times around the motor pulley and attached at each end of the large partial pulley. In one instantiation, the output of the digital-to-analog (D/A) converter from the microprocessor is passed through a current amplifier that gives a current through the motor that is proportional

to the D/A voltage. This gives direct control of applied torque on the motor. The resulting system has a force felt at the driving point that is proportional to the output voltage during static operation. When the system is moving, the force applied to the operator may differ due to human and device dynamics.

42.3 Haptic Rendering

Haptic rendering (in impedance systems) is the process of computing the force required by contacts with virtual objects based on measurements of the operator's motion. This section describes haptic rendering for virtual environments. Haptic feedback for teleoperators is described in Chap. 43.

An important property of haptic systems is that their timing constraints are quite severe. To illustrate this point, tap a pencil on the table top. You hear a sound which is an audio representation of the contact dynamics between pencil tip and table top. Tactile receptors in the human fingers are known to respond up to 10 kHz [42.1]. To realistically render this type of contact between hard surfaces would require response well into the audio range of frequencies (up to 20 kHz) and thus sampling times of around 25 μs. Even with specialized designs, haptic devices do not have this bandwidth, and such high fidelity is not usually a goal of haptic rendering. Achieving stability in any type of hard contact requires very high sampling rates. In practice, most haptic simulation systems are implemented with at least 1000 Hz sampling rate. This can be reduced to the low hundreds of Hertz if the virtual environment is limited to soft materials.

Basic Haptic Rendering

The computational process of haptic rendering can be formulated into the following seven sequential steps for each cycle (Fig. 42.7). The rendering cycle must typically be completed in under 1 ms for stability and realism:

1. Sensing (Sect. 42.2.2)
2. Kinematics
3. Collision detection
4. Determining surface point
5. Force calculation
6. Kinematics
7. Actuation (Sect. 42.2.3).

Kinematics

The position and velocity measurements acquired by sensors are typically in joint space. These must be converted through the forward kinematics model and the Jacobian matrix (Chap. 2) to the Cartesian position and velocity of the operator's hand or fingertip. In some applications the operator is virtually holding a tool or object whose shape is represented in the virtual environment but whose position and orientation are determined by the operator. In whose others, the operator's fingertip or hand is represented by a point which makes only point contact with objects in the virtual environment (VE). We refer to a virtual handheld object as a virtual tool and, following [42.39], we refer to the single end-point as the haptic interaction point (HIP).

Collision Detection

For the point contact case, the collision detection software must determine if the position of the HIP at the current instant of time represents contact with a virtual object. In practice this usually means to determine if the HIP is penetrating or inside the object surface. The object surface is represented by a geometric model such as polygons or splines.

Although there is an extensive literature on collision detection in computer graphics, there are unique aspects of the collision detection problem for haptics. In particular, speed of computation is paramount and worst-case speed, as opposed to average speed, is what counts. Solutions that evaluate in constant time are pre-

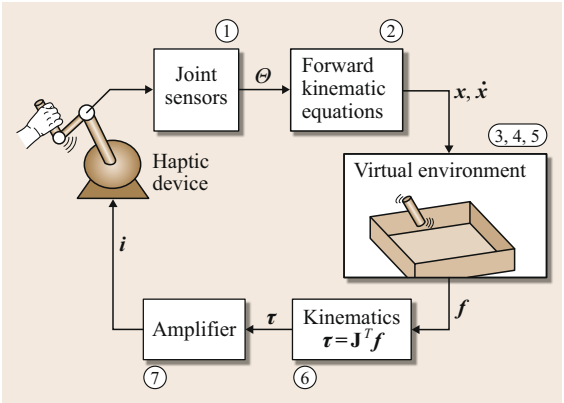


Fig. 42.7 Schematic diagram of the haptic rendering cycle for an impedance haptic display system. Virtual object moves in the virtual environment according to operator's displacement of the haptic device. Joint displacements (Θ) sensed in the device (1) are processed through kinematics (2), collision detection (3), surface point determination (4), force calculation (5), kinematics (6), and actuation (7)

Avidbots Ex1016
Page 1101 of 2154

ferred. Section 42.3.1 addresses collision detection and haptic rendering for complex environments.

If the HIP is found to be outside all objects, then a force of zero is returned.

Determining Surface Point

Once it is determined that the HIP is inside a surface, the force to be displayed to the operator must be computed. Many researchers have used the idea of a virtual spring connecting the HIP to the nearest point on the surface as a model of interpenetration and force generation [42.39–41]. Basdogan and Srinivasan named this point the intermediate haptic interaction point (IHIP). However all of these authors realized that the closest surface point is not always the most faithful model of contact. For example, as the HIP moves laterally below the top surface of a cube (Fig. 42.8), eventually it becomes close enough to the edge that the closest surface point becomes the side of the cube. In this situation, the algorithm needs *memory* to keep the IHIP on the top surface and generate an upward force at all times or the operator is suddenly ejected out the side of the cube.

Force Calculation

Force is commonly computed by using the spring model (Hooke's Law)

$$\boldsymbol{f} = k\boldsymbol{x}\,, \tag{42.3}$$

where $\boldsymbol{x}$ is the vector from the HIP to the IHIP, and $k > 0$. When k is sufficiently large, the object surface will feel like a wall perpendicular to $\boldsymbol{x}$. This *virtual wall*, or impedance surface, is a fundamental building block of most haptic virtual environments. Because the virtual wall is only displayed if a collision between the HIP and virtual object is detected, the wall is a unilateral constraint, governed by a nonlinear switching condition. As will be described in the following section, haptic virtual environments with complex geometries are often formed using a polygonal mesh in which each polygon is essentially a virtual wall. A virtual surface may also be allowed to deform globally, while the local interaction with the operator is governed by a virtual wall. *Virtual fixtures*, which are often constructed from virtual walls, can be overlaid on haptic feedback teleoperators to assist the operator during a teleoperated task (Chap. 43).

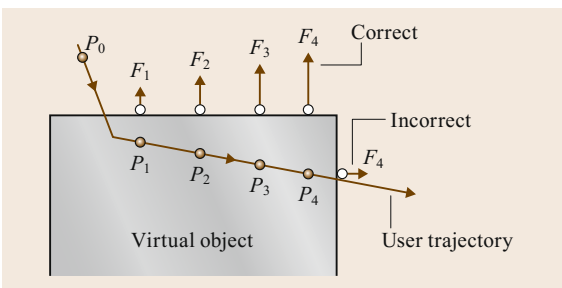


Fig. 42.8 Illustration of subtle aspects of haptic rendering of contact force. The operator fingertip trajectory enters object surface moving down and to right. Haptic interaction points (HIP) are shown at times 1–4 (*solid circles*, P_0–P_4). Intermediate haptic interaction points (IHIP) are shown when the HIP is inside the object (*open circles*). At position P_4, the algorithm must not render force based on the closest surface point or the operator will be ejected from the side of the object (feeling unnatural force tangential to the top surface)

The pure stiffness model described above can be augmented to provide other effects, particularly through the use of the *virtual coupling* described in Sect. 42.3.2. Damping can be added perpendicular or parallel to the surface. In addition, Coulomb or other nonlinear friction may be displayed parallel to the surface. To provide a more realistic display of hard surfaces, vibrations can also be displayed open loop at the moment of collision between the HIP and the surface, as is described in Sect. 42.5.

Kinematics

The computed force in Cartesian space must then be transformed into torques in the actuator space. Typically the calculation is

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\boldsymbol{f}\,, \tag{42.4}$$

where $\boldsymbol{\tau}$ is the torque command to the actuators, $\boldsymbol{f}$ is the desired force vector, and $\mathbf{J}^{\mathrm{T}}$ is the transpose of the haptic device Jacobian matrix (Chap. 2). If the haptic device has no dynamics and the actuators are perfect, the exact desired force is displayed to the operator. However, real device dynamics, time delays, and other nonidealities result in applied operator forces that differ from the desired forces.

42.3.1 Rendering Complex Environments

Today's computer power is sufficient that a variety of relatively simple algorithms can effectively render haptics for simple virtual environments, say consisting of a handful of simple geometric primitives such as spheres, cubes, and planes. However, the challenge is to scale these algorithms to complex environments such as those we are used to seeing in computer graphic renderings consisting of 10^5–10^7 polygons. A variety of approaches have been tried in the literature for efficient rendering of complex scenes. *Zilles* and *Salisbury* [42.40] found planar constraints due to

the nearby surface polygons and solved for the closest IHIP point using Lagrange multipliers. *Ruspini* and *Khatib* [42.42] added force shading and friction models. *Ho* et al. [42.39] used a hierarchy of bounding spheres to determine the initial contact (collision) point, but thereafter searched neighboring surfaces, edges, and vertices (referred to as geometric primitives) of the current contacted triangle to find the closest point for the IHIP. *Gregory* et al. [42.43] imposed a discretization of three-dimensional (3-D) space onto the hierarchy to speed up the detection of initial contact with the surface. *Johnson* et al. [42.44] performed haptic rendering on moving models based on local extrema in distance between the model controlled by the haptic device and the rest of the scene. *Lin* and *Otaduy* [42.45, 46] used level-of-detail representations of the objects for performing multiresolution collision detection, with the goal of satisfying real-time constraints while maximizing the accuracy of the computed proximity information.

Alternative algorithms and efficiencies can be derived by not representing the object surface as polygons. *Thompson* and *Cohen* [42.47] derived the mathematics for computing the interpenetration depth for surfaces directly from non-uniform rational B-spline (NURBS) models. *McNeely* et al. [42.18] took the extreme approach of voxelizing space at the millimeter scale. Each voxel contained a precomputed normal vector, stiffness properties, etc. 1000 Hz rendering was achieved with very complex CAD models containing millions of polygons (in the graphical equivalent representation), but large amounts of memory and precomputation are required. The performance of this algorithm is sufficiently high that it can be used to render hundreds of contact points simultaneously. This allows the operator to hold an arbitrarily shaped tool or object. The tool/object is populated by points surrounding its surface and the resultant force and moment on the tool is the sum of interaction forces computed on all of the surface points.

For surgical simulation, researchers have focused on the modeling and haptic rendering of the interaction between surgical instruments and organs. Researchers have attempted to model virtual tissue behavior in a wide variety of ways, which can be broadly classified as:

1. Linear elasticity based
2. Nonlinear (hyperelastic) elasticity-based finite-element (FE) methods
3. Other techniques that are not based on FE methods or continuum mechanics.

While most conventional linear and nonlinear FE algorithms cannot be run in real time, methods such as preprocessing can allow them to run at haptic rates [42.48]. Many researchers rely on data acquired from real tissues to model organ deformation and fracture accurately. Major challenges in this field include the modeling of connective tissue supporting the organ, friction between instruments and tissues, and topological changes occurring during invasive surgical procedures.

42.3.2 Virtual Coupling

So far we have rendered forces by computing the length and direction of a virtual spring and applying Hooke's Law (42.3). This spring is a special case of a *virtual coupling* [42.41] between the HIP and the IHIP. The virtual coupling is an abstraction of the interpenetration model of force rendering. Instead of viewing the objects as compliant, we assume them to be rigid but connect them to the operator through a virtual spring. This imposes an effective maximum stiffness (that of the virtual coupling).

Problems with stable contact rendering (Sect. 42.4) often require more sophisticated virtual couplings than a simple spring. For example, damping can be added, generalizing the force rendering model of (42.3) to

$$\boldsymbol{f} = k\boldsymbol{x} + b\dot{\boldsymbol{x}} \,. \tag{42.5}$$

The parameters k and b can be empirically tuned for stable and high-performance operation. More-formal design methods for the virtual coupling are covered in Sect. 42.4.

42.4 Control and Stability of Force Feedback Interfaces

Introduction to the Problem

The haptic rendering system depicted in Fig. 42.7 is a closed-loop dynamical system. It is a challenge to render realistic contact forces yet retain the stable behavior of human-environment contact in the natural world. Instability in haptic interfaces manifests itself as buzzing, bouncing, or even wildly divergent behavior. The worst case for impedance devices is during attempted contact with stiff objects. Empirically, instability is frequently encountered when developing haptic interfaces with stiff virtual objects, but this instability can be eliminated by reducing the stiffness of the virtual object or by the operator making a firmer grasp on the haptic device.

Problem Description in Classical Control Terms

Although linear theory is of very limited use in haptic control applications, it can be applied to a basic analysis of the factors affecting instability [42.49]. Such a highly simplified model of an impedance device is shown in Fig. 42.9. $G_1(s)$ and $G_2(s)$ represent dynamics of the haptic device for both operator position sensing and force display respectively. Assume that the virtual environment and human operator/user (HO) can each be represented by a linear impedance such as

$$Z_{\mathrm{VE}} = \frac{F_{\mathrm{VE}}(s)}{X_{\mathrm{VE}}(s)} , \tag{42.6}$$

$$Z_{\mathrm{HO}} = \frac{F_{\mathrm{HO}}(s)}{X_{\mathrm{HO}}(s)} . \tag{42.7}$$

Then the loop gain of the closed-loop system from the human operator and back again is

$$G_l(s) = G_1(s)G_2(s)\frac{Z_{\mathrm{VE}}(s)}{Z_{\mathrm{HO}}(s)} . \tag{42.8}$$

Stability in the classical sense is assessed by applying the magnitude and phase criteria of Nyquist to $G_l(s)$. Increasing Z_{VE} (corresponding to stiffer or heavier virtual objects) increases the magnitude of $G_l(s)$ and thus destabilizes the system while a firmer grasp by the human operator, which increases the magnitude of Z_{HO}, has a stabilizing effect. Similar arguments apply to phase shifts that might be present in any part of the system.

Limitations of Linear Theory

Although the model of Fig. 42.9 illustrates some qualitative features of haptic interface stability, linear continuous time theory is of little use in designing methods to stabilize the loop. Interesting virtual environments are nonlinear. Furthermore, they can rarely be linearized because applications often simulate discontinuous contact, for example, between a stylus in free space and a hard surface. A second feature is digital implementation, which introduces sampling and quantization – both of which have significant effects.

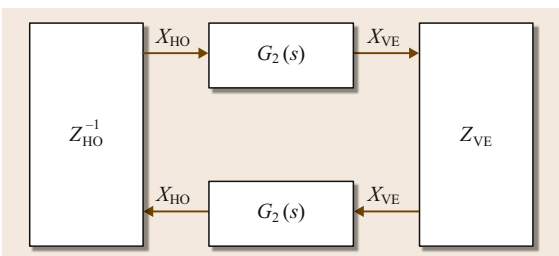


Fig. 42.9 Highly simplified linear model of haptic rendering to highlight some stability issues

Sampling

Colgate et al. [42.41] incorporated consideration of discrete time sampling behavior in the stability analysis. They considered the problem of implementing a virtual wall of stiffness

$$H(z) = K + B\frac{z-1}{Tz} , \tag{42.9}$$

where K is the virtual wall stiffness, B is the virtual wall damping coefficient, z is the z-transform variable, and T is the sampling time. They further modeled the haptic device (HD) in continuous time as

$$Z_{\mathrm{HD}}(s) = \frac{1}{ms+b} , \tag{42.10}$$

where m and b are the mass and damping of the haptic device, respectively. They derived the following condition for passivity of the device

$$b > \frac{KT}{2} + |B| , \tag{42.11}$$

showing a significant stabilizing effect of high sampling rates and also of high mechanical damping in the haptic device.

Quantization

Additional factors include delays due to numerical integration schemes and quantization. These contributing factors to instability have been termed *energy leaks* by *Gillespie* and *Cutkosky* [42.50].

Passivity

Interesting virtual environments are always nonlinear and the dynamic properties of a human operator are important. These factors make it difficult to analyze haptic systems in terms of known parameters and linear control theory. One fruitful approach is to use the idea of passivity to guarantee stable operation. Passivity is a sufficient condition for stability, and is reviewed more completely in Chap. 43 on telerobotics. There are many similarities between the control of haptic interfaces and bilateral teleoperation.

The major problem with using passivity for design of haptic interaction systems is that it is overly conservative, as shown in [42.35]. In many cases, performance can be poor if a fixed damping value is used to guarantee passivity under all operating conditions. *Adams* and *Hannaford* derived a method of virtual coupling design from two-port network theory which applied to all causality combinations and was less conservative than passivity based design [42.51]. They were able to derive optimal virtual coupling parameters using a dynamic model of the haptic device and by satisfying Lewellyn's *absolute stability criterion*, an inequality

Avidbots Ex1016
Page 1104 of 2154

composed of terms in the two-port description of the combined haptic interface and virtual coupling system. *Miller* et al. derived another design procedure which extended the analysis to nonlinear environments and extracted a damping parameter to guarantee stable operation [42.52–54].

42.5 Other Types of Haptic Interfaces

While kinesthetic (force feedback) haptic interfaces have the closest relationship to robotics, there are a variety of other types of haptic interfaces which are usually classified as tactile displays. Tactile displays are used to convey force, contact and shape information to the skin. They purposely stimulate cutaneous receptors, with little effect on kinesthetic sensation. This is in contrast to kinesthetic displays, which must inherently provide some cutaneous sensations through physical contact with a tool or thimble, but whose primary output is force or displacement to the limbs and joints. Tactile displays are usually developed for a specific purpose, such as display of contact events, contact location, slip/shear, texture, and local shape. Special purpose tactile displays can be targeted at different types of cutaneous receptors, each with its own frequency response, receptive field, spatial distribution, and sensed parameter (e.g., local skin curvature, skin stretch, and vibration) and each receptor type is associated with different exploratory procedures described in Sect. 42.1.1.

In contrast, tactile displays that render contact information for virtual reality or teleoperation have proven far more challenging. Accurate recreation of the local shape and pressure distribution at each fingertip requires a dense array of actuators. Devices specifically aimed at tactile perception are an active area of research but most have not reached the stage of applications or commercial distribution, with the notable exception of Braille displays for the blind. In this section, we describe the various types of tactile displays, their design considerations and specialized rendering algorithms, and applications.

42.5.1 Vibrotactile Feedback

Vibrotactile feedback is a popular method of providing tactile feedback. It can be used as a stand-alone method for haptic feedback or as an addition to a kinesthetic display. Vibrating elements, such as piezoelectric materials and small voice-coil motors, are lighter than the actuators used in kinesthetic devices, and can often be added to kinesthetic devices with little impact on existing mechanisms. In addition, high-bandwidth kinesthetic displays can be programmed to display open-loop vibrations through their normal actuators. The sensitivity for human vibration sensing ranges from DC to over 1 kHz, with peak sensitivity around 250 Hz.

We first consider the use of vibrations to convey impact or contact events – a technique that straddles kinesthetic and tactile feedback. When humans touch an environment, fast-acting sensors embedded in the skin record the minute vibrations occurring from this interaction. As described in Sect. 42.3, conventional approaches to haptic display usually consist of designing a virtual model with simple geometry, then using a first-order stiffness control law to emulate a surface. However, such first-order models often lack the realism of higher-order effects such as impact. With common haptic rendering algorithms, surfaces feel squishy or unrealistically smooth. One solution to improving the realism of such environments is to add higher-order effects such as textures and contact vibrations. These effects can use a library of surface models based on ad hoc analytical descriptions [42.55], which are sometimes tuned using qualitative operator feedback, physical measurements (reality-based models created from empirical data) [42.56, 57], or a combination of the two [42.58]. At the instant collision is detected between the HIP and a surface of the virtual object, the appropriate waveform is called out of the library, scaled according to the context of the motion (such as velocity or acceleration), and played open loop through an actuator. That actuator may be the same one simultaneously displaying lower-frequency force information, as shown in Fig. 42.10, or it might be a separate transducer. *Kuchenbecker* et al. [42.59] considered the dynamics of the haptic device to display the most accurate vibration waveforms possible, and compared a number of different vibration waveform generation techniques meant to convey impact superimposed on force feedback in virtual environments. Most of the vibration feedback methods performed similarly in terms of realism, and they were also significantly more realistic than conventional force feedback alone.

Vibration feedback can also be used to provide information about patterned textures, roughness, and other phenomena that have clear vibratory signals. This type of vibration feedback is often termed *vibrotactile* feedback. In teleoperated environments, *Kontarinis* and *Howe* [42.60] showed that damaged ball bear-

ings could be identified through vibration feedback, and *Dennerlein* et al. [42.61] demonstrated that vibration feedback improved performance over no haptic feedback, for a telemanipulation task inspired by undersea field robotics. In these teleoperator systems, vibration-sensitive sensors such as accelerometers and piezoelectric sensors are used to pick up and, in most cases, directly provide the vibration signals as inputs to a vibrotactile actuator. In virtual environments, *Okamura* et al. [42.62] displayed vibrations modeled based on textures and puncture of a membrane. Similar to the event-based haptics above, the vibration waveforms were modeled based on earlier experiments and played open loop during interaction with the virtual environment.

Finally, vibration feedback has been used as a method of sensory substitution, to convey direction, attention, or other information, e.g., [42.63–65]. In this case, the strength and clarity of the signal, not realism, is the goal. Vibration frequencies near the peak human sensitivity are most effective, and there exist commercially available vibrotactile actuators (or *tactors*) suitable for such applications, e.g., Engineering Acoustic, Inc.'s C2 Tactor [42.66]. Elements within an array of tactors can be selectively turned on and off to evoke the sensory saltation phenomenon, in which a pattern of brief pulses on a series of tactors is perceived not as successive taps at different locations, but as a single tap that is traveling or hopping over the skin.

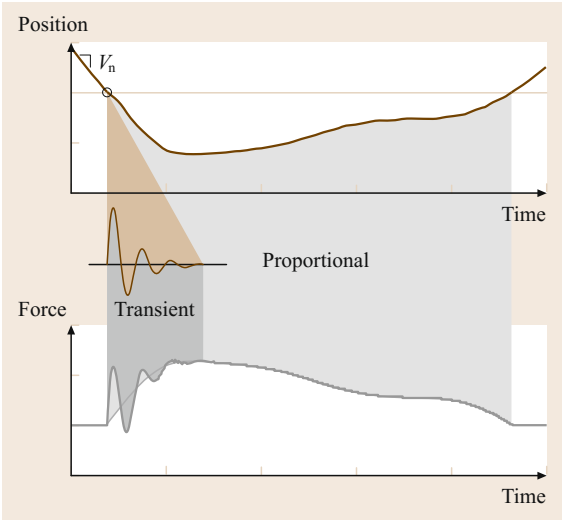


Fig. 42.10 Event-triggered open-loop force signals superimpose on traditional penetration-based feedback forces, providing vibration feedback that improves the realism of hard surfaces in virtual environments (after [42.59] with permission)

42.5.2 Contact Location, Slip, and Shear Display

In early work related to robotic dexterous manipulation, it was found that knowledge of the contact point between a robot hand and a grasped object is essential for manipulation. Without this knowledge, the robot will easily drop the object due to rapid accumulation of grasp errors. While many robotics researchers and some companies have developed tactile array sensors capable of measuring contact location, pressure distribution, and local object geometry (Chap. 28), practical methods for display of this information to the human operator of a virtual or teleoperated environment have proven much more difficult. We begin our discussion of contact display by considering contact location, slip, and shear display, which have the common goal of displaying the motion of a single area of contact relative to the skin (almost invariably on a finger). Arrays of pins that rise and fall to create a pressure distribution on the skin have been the most popular method to date for displaying contact information, but we will address those designs in the following section on local shape, since their primary advantage is the display of spatially distributed information. Instead, we will focus here on tactile devices that are designed to specifically address the problem of contact location and motion.

As an example of contact location display, *Provancher* et al. [42.67] developed a system that renders the location of the contact centroid moving on the user's fingertip. The tactile element is a free-rolling cylinder that is normally suspended away from the fingertip, but comes into contact with the skin when the operator pushes on a virtual object. The motion of the cylinder over the skin is controlled by sheathed push-pull wires. This allows the actuators to be placed remotely, creating a lightweight, thimble-sized package that can be unobtrusively mounted on a kinesthetic haptic device. An experiment demonstrated that human operators performed similarly during real manipulation and virtual manipulation (using the tactile display) in an object curvature discrimination task. In addition, operators were able to use the device to distinguish between manipulations of rolling and anchored but rotating virtual objects.

42.5.3 Slip and Shear

Humans use slip and incipient slip widely during manipulation tasks [42.68]. To reproduce these sensations for experiments to characterize human slip sensation, researchers have created stand-alone 1-DOF slip displays [42.69–71]. *Webster* et al. [42.71] created a 2-DOF tactile slip display, which uses an actuated ro-

tating ball positioned under the user's fingertip. The lightweight, modular tactile display can be attached to a multi-DOF kinesthetic interface and used to display virtual environments with slip. Experimental results demonstrate that operators complete a virtual manipulation task with lower applied forces using combined slip and force feedback in comparison with conventional force feedback alone. Skin stretch can also be integrated with a slip display to provide information about pre-slip conditions. For example, *Tsagarakis* et al. [42.72] developed a lightweight device that uses a V configuration of miniature motors to provide sensations of relative lateral motion (direction and velocity) onto the operator's fingertips. Generation of two-dimensional (2-D) slip/stretch is achieved by coordinating the rotational speed and direction of the two motors.

In terms of tactile device kinematics, slip and shear displays can be quite similar. However, the goal of shear, or skin stretch, displays is to maintain a no-slip condition such that shear forces/motions can be accurately controlled with respect to the human operator, typically on the finger. The use of shear (tangential skin stretch) is motivated by perceptual experiments demonstrating that the human fingerpad is more sensitive to tangential displacement compared to normal displacement [42.73]. A variety of tangential skin stretch devices have been designed, with applications in wearable haptics, high-fidelity tactile rendering, and teleoperation. *Hayward* and *Cruz-Hernandez* [42.74] developed a tactile device consisting of closely packed piezoelectric actuators, which generates a programmable stress field within the fingerpad. Single skin-stretch *tactors* have also been used to convey two-dimensional directional information (for navigation) [42.75] and to enhance perception of virtual environments (e.g., friction) [42.76] alone or in combination with kinesthetic haptic interfaces. A similar approach has been used for 3-DOF skin stretch, in combination with cutaneous normal force [42.77, 78]. In a different form factor, researchers have developed rotational skin stretch devices that can act as a substitute for natural proprioceptive feedback [42.79, 80].

42.5.4 Local Shape

Most tactile devices for local shape display consist of an array of individual pin elements that move normal to the surface. Often, a layer of elastic material is used to cover the pins so that the operator contacts a smooth surface rather than the pins directly. Other systems use individual elements that move laterally, and some substitute electrodes for moving parts to form an array of electrocutaneous elements. A number of researchers have used psychophysical and perceptual experimental results to define design parameters such as number of pins, spacing, and amplitude of pin-based tactile displays. A commonly used metric is the two-point discrimination test, which defines the minimum distance between two contact points on the skin at which they are perceived as two, rather than one point. This discrimination limit varies widely for skin on different parts of the body, with the fingertips having one of the smallest (usually cited as less than 1 mm, although this depends on the shape and size of the contacts) [42.81, 82]. *Moy* et al. [42.83] quantified several perceptual capabilities of the human tactile system based on predicted subsurface strain and psychophysical experiments that measured amplitude resolution, the effects of shear stress, and the effects of viscoelasticity (creep and relaxation) on tactile perception for static touch. They found find that 10% amplitude resolution is sufficient for a teletaction system with a 2 mm elastic layer and 2 mm tactor spacing. A different type of experiment examines the kind of tactile information relevant to a particular application. For example, *Peine* and *Howe* [42.84] found that sensed deformation of the fingerpad, and not changes in pressure distribution, were responsible for localizing lumps in a soft material, such as a tumor in tissue.

We will now highlight a few distinctive designs of array-type tactile displays. A number of actuator technologies have been applied to create tactile arrays, including piezoelectric, shape-memory alloy (SMA), electromagnetic, pneumatic, electrorheological, microelectromechanical system (MEMS), and electrotactile. Further reading on tactile display design and actuation is available in review papers, including [42.85–90].

We will first consider two ends of the spectrum in complexity/cost of the pin-based approach. *Killebrew* et al. [42.91] developed a 400-pin, 1 cm^2 tactile stimulator to present arbitrary spatiotemporal stimuli to the skin for neuroscience experiments. Each pin is under independent computer control and can present over 1200 stimuli per minute. While not practical for most haptic applications due to the size and weight of the actuation unit, it is the highest-resolution tactile display built to date and can be used to evaluate potential designs for lower-resolution displays. *Wagner* et al. [42.92] created a 36-pin, 1 cm^2 tactile shape display that uses commercially available radio-controlled (RC) servomotors. The display can represent maximum frequencies of 7.5−25 Hz, pending on the amount of pin deflection, and is shown in Fig. 42.11. *Howe* et al. [42.93, 94] have also explored the use of shape-memory alloys for pin actuation, for the application of remote palpation.

In contrast to pins that move normal to the surface, recent tactile array designs have incorporated

pins that move laterally. First introduced by *Hayward* and *Cruz-Hernandez* [42.95], the most recent compact, lightweight, modular design [42.96] uses a 6×10 piezo bimorph actuator array with a spatial resolution of $1.8\,\text{mm} \times 1.2\,\text{mm}$. The force of the individual actuators provides sufficient skin pad motion/stretch to excite mechanoreceptors [42.97]. A pilot test demonstrated that subjects could detect a virtual line randomly located on an otherwise smooth virtual surface, and the device has also been tested as a Braille display [42.98]. Another lateral stretch display and its evaluation is described in [42.99]. Other novel approaches to tactile display include sending small currents through the skin or tongue using an electrocutaneous array [42.100] and the application of air pressure to stimulate only superficial mechanoreceptors [42.101].

42.5.5 Surface Displays

In recent years, the concept of surface displays, which modulate surface friction in order to display changing shear forces as a user moves the finger over the surface. One example device that uses slip and friction to display compelling tactile sensations is the TPaD (tactile pattern display) [42.102]. Ultrasonic frequency, low amplitude vibrations of a flat plate create a film of air between the plate and a human finger touching the plate, thereby reducing friction. The 33 kHz vibration of the plate cannot be perceived by the human. The amount of friction reduction varies with vibration amplitude, allowing indirect control of shear forces on the finger during active exploration. Finger position and velocity feedback enables haptic rendering of spatial texture sensations. This work has been expanded into a variety of different surface displays, including devices that generate active forces [42.103] and those that use electrostatic forces to modulate friction [42.104, 105].



Fig. 42.11 **(a)** A low-cost 36-pin tactile display using RC servomotor actuation. **(b)** A closeup of the 6×6 display, showing a sine wave grating (after [42.92] with permission)

42.5.6 Temperature

Because the human body is typically warmer than objects in the environment, thermal perceptions are based on a combination of thermal conductivity, thermal capacity, and temperature. This allows us to infer not only temperature difference, but also material composition [42.106]. Most thermal display devices are based on thermoelectric coolers, also known as Peltier heat pumps. Thermoelectric coolers consist of a series of semiconductor junctions connected electrically in series and thermally in parallel. The thermoelectric cooler is designed to pump heat from one ceramic faceplate to the other, but if used in reverse, a temperature gradient across the device produces a proportional potential; as a measure of relative temperature change. The designs of haptic thermal displays mostly use off-the-shelf components, and their applications are typically straightforward, enabling identification of objects in a virtual or teleoperated environment by their temperature and thermal conductivity.

Ho and *Jones* [42.107] provide a review of haptic temperature display, as well as promising results suggesting that a thermal display is capable of facilitating object recognition when visual cues are limited. Although numerous systems have integrated thermal display with other types of haptic display, the Data Glove Input System designed by *Caldwell* et al. [42.108, 109] was one of the first to do so. Their haptic interface provides simultaneous force, tactile, and thermal feedback. The Peltier device used for thermal display contacts the dorsal surface of the index finger. Subjects achieved a 90% success rate in identifying materials such as a cube of ice, a soldering iron, insulating foam, and a block of aluminum, based only on thermal cues. The study of human temperature perception is particularly interesting, including issues such as spatial summation and the psychological relevance of temperature display. For example, in prosthetic limbs, temperature display

may be useful not only for practical reasons such as safety and material identification, but also for reasons of personal comfort, such as feeling the warmth of a loved one's hand.

42.6 Conclusions and Further Reading

Haptic technology, which attempts to provide compelling sensations to human operators in virtual and teleoperated environments, is a relatively new, but fast-growing and dynamic area of research. The field relies not only on fundamental foundations from robotics and control theory, but also on fields in the human sciences, particularly neuroscience and psychology. To date, commercial success of haptics has been in the areas of entertainment, medical simulation, and design, although novel devices and applications are regularly appearing.

There exist many books on the topic of haptic technology, most of them compendiums from workshops or conferences on the subject. One of the earliest books on haptics, by *Burdea* [42.110], provides a thorough review of applications and haptic devices up to 1996. A book specifically focused on haptic rendering, designed for potential use as a textbook, has been edited by *Lin* and *Otaduy* [42.111]. In addition, we recommend the following useful articles: *Hayward* and *MacLean* [42.112, 113] describe the fundamentals of constructing experimental haptic devices of modest complexity, the software components needed to drive them, and the interaction design concepts important to creating usable systems. *Hayward* et al. [42.114] also provide a tutorial on haptic devices and interfaces. *Salisbury* et al. [42.115] describe the basic principles of haptic rendering. *Hayward* and *MacLean* [42.116] describes a number of tactile illusions that can inspire creative solutions for haptic interface design. *Robles-De-La-Torre* [42.117] underscores the importance of haptics with compelling examples of humans who have lost the sense of touch.

Finally, there exist two journals that are specific to the field of haptics: *Haptics-e* [42.118] and *IEEE Transactions on Haptics* (first issue expected 2008). Several conferences are specifically devoted to haptics: Eurohaptics and the Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems are held separately in even years, and on the odd years become a single conference, World Haptics. The IEEE (Institute of Electrical and Electronics Engineers) technical committee on haptics [42.119] provides information about relevant publication forums.

References

42.1 K.B. Shimoga: A survey of perceptual feedback issues in dexterous telemanipulation. I. Finger force feedback, Proc. Virtual Real. Annu. Int. Symp. (1993) pp. 263–270

42.2 M.A. Srinivasan, R.H. LaMotte: Tactile discrimination of shape: responses of slowly and rapidly adapting mechanoreceptive afferents to a step indented into the monkey fingerpad, J. Neurosci. **7**(6), 1682–1697 (1987)

42.3 R.H. LaMotte, R.F. Friedman, C. Lu, P.S. Khalsa, M.A. Srinivasan: Raised object on a planar surface stroked across the fingerpad: Responses of cutaneous mechanoreceptors to shape and orientation, J. Neurophysiol. **80**, 2446–2466 (1998)

42.4 R.H. LaMotte, J. Whitehouse: Tactile detection of a dot on a smooth surface: Peripheral neural events, J. Neurophysiol. **56**, 1109–1128 (1986)

42.5 R. Hayashi, A. Miyake, H. Jijiwa, S. Watanabe: Postureal readjustment to body sway induced by vibration in man, Exp. Brain Res. **43**, 217–225 (1981)

42.6 G.M. Goodwin, D.I. McCloskey, P.B.C. Matthews: The contribution of muscle afferents to kinesthesia shown by vibration induced illusions of movement an the effects of paralysing joint afferents, Brain **95**, 705–748 (1972)

42.7 G.S. Dhillon, K.W. Horch: Direct neural sensory feedback and control of a prosthetic arm, IEEE Trans. Neural Syst. Rehabil. Eng. **13**(4), 468–472 (2005)

42.8 G.A. Gescheider: *Psychophysics: The Fundamentals* (Lawrence Erlbaum, Hillsdale 1985)

42.9 L.A. Jones: Perception and control of finger forces, Proc. ASME Dyn. Syst. Control Div. (1998) pp. 133–137

42.10 R. Klatzky, S. Lederman, V. Metzger: Identifying objects by touch, An 'expertt system', Percept. Psychophys. **37**(4), 299–302 (1985)

42.11 S. Lederman, R. Klatzky: Hand movements: A window into haptic object recognition, Cogn. Psychol. **19**(3), 342–368 (1987)

42.12 S. Lederman, R. Klatzky: Haptic classification of common objects: Knowledge-driven exploration, Cogn. Psychol. **22**, 421–459 (1990)

42.13 O.S. Bholat, R.S. Haluck, W.B. Murray, P.G. Gorman, T.M. Krummel: Tactile feedback is present

Avidbots Ex1016
Page 1109 of 2154

during minimally invasive surgery, J. Am. Coll. Surg. **189**(4), 349–355 (1999)
42.14 C. Basdogan, S. De, J. Kim, M. Muniyandi, M.A. Srinivasan: Haptics in minimally invasive surgical simulation and training, IEEE Comput. Graph. Appl. **24**(2), 56–64 (2004)
42.15 P. Strom, L. Hedman, L. Sarna, A. Kjellin, T. Wredmark, L. Fellander-Tsai: Early exposure to haptic feedback enhances performance in surgical simulator training: A prospective randomized crossover study in surgical residents, Surg. Endosc. **20**(9), 1383–1388 (2006)
42.16 A. Liu, F. Tendick, K. Cleary, C. Kaufmann: A survey of surgical simulation: Applications, technology, and education, Presence Teleop. Virtual Environ. **12**(6), 599–614 (2003)
42.17 R.M. Satava: Accomplishments and challenges of surgical simulation, Surg. Endosc. **15**(3), 232–241 (2001)
42.18 W.A. McNeely, K.D. Puterbaugh, J.J. Troy: Six degree-of-freedom haptic rendering using voxel sampling, Proc. SIGGRAPH 99 (1999) pp. 401–408
42.19 Geomagic Touch: http://www.geomagic.com
42.20 T.H. Massie, J.K. Salisbury: The phantom haptic interface: A device for probing virtual objects, Proc. ASME Dyn. Syst. Contr. Div., Vol. 55 (1994) pp. 295–299
42.21 Novint Technologies: http://www.novint.com
42.22 Chai3D: http://www.chai3d.org
42.23 M.C. Cavusoglu, D. Feygin, F. Tendick: A critical study of the mechanical and electrical properties of the PHANToM haptic interface and improvements for high-performance control, Presence **11**(6), 555–568 (2002)
42.24 R.Q. van der Linde, P. Lammerste, E. Frederiksen, B. Ruiter: The HapticMaster, a new high-performance haptic interface, Proc. Eurohaptics Conf. (2002) pp. 1–5
42.25 T. Yoshikawa: Manipulability of robotic mechanisms, Int. J. Robotics Res. **4**(2), 3–9 (1985)
42.26 J.K. Salibury, J.T. Craig: Articulated hands: Force control and kinematics issues, Int. J. Robotics Res. **1**(1), 4–17 (1982)
42.27 P. Buttolo, B. Hannaford: Pen based force display for precision manipulation of virtual environments, Proc. Virtual Reality Annu. International Symposium (VRAIS) (1995) pp. 217–225
42.28 P. Buttolo, B. Hannaford: Advantages of actuation redundancy for the design of haptic displays, Proc. ASME 4th Annu. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst., Vol. 57-2 (1995) pp. 623–630
42.29 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)
42.30 S. Venema, B. Hannaford: A probabilistic representation of human workspace for use in the design of human interface mechanisms, IEEE Trans. Mechatron. **6**(3), 286–294 (2001)
42.31 H. Yano, M. Yoshie, H. Iwata: development of a non-grounded haptic interface using the gyro effect, Proc. 11th Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (2003) pp. 32–39
42.32 C. Swindells, A. Unden, T. Sang: TorqueBAR: an ungrounded haptic feedback device, Proc. 5th Int. Conf. Multimodal Interface (2003) pp. 52–59
42.33 Immersion Corporation: CyberGrasp – Groundbreaking haptic interface for the entire hand, http://www.immersion.com/3d/products/cyber_grasp.php (2006)
42.34 C. Richard, M.R. Cutkosky: Contact force perception with an ungrounded haptic interface, Proc. ASME Dyn. Syst. Control Div. (1997) pp. 181–187
42.35 J.J. Abbott, A.M. Okamura: Effects of position quantization and sampling rate on virtual-wall passivity, IEEE Trans. Robotics **21**(5), 952–964 (2005)
42.36 S. Usui, I. Amidror: Digital low-pass differentiation for biological signal processing, IEEE Trans. Biomed. Eng. **29**(10), 686–693 (1982)
42.37 P. Bhatti, B. Hannaford: Single chip optical encoder based velocity measurement system, IEEE Trans. Contr. Syst. Technol. **5**(6), 654–661 (1997)
42.38 A.M. Okamura, C. Richard, M.R. Cutkosky: Feeling is believing: Using a force-feedback joystick to teach dynamic systems, ASEE J. Eng. Educ. **92**(3), 345–349 (2002)
42.39 C.H. Ho, C. Basdogan, M.A. Srinivasan: Efficient point-based rendering techniques for haptic display of virtual objects, Presence **8**, 477–491 (1999)
42.40 C.B. Zilles, J.K. Salisbury: A constraint-based god-object method for haptic display, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 146–151
42.41 J.E. Colgate, M.C. Stanley, J.M. Brown: Issues in the haptic display of tool use, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 140–145
42.42 D. Ruspini, O. Khatib: Haptic display for human interaction with virtual dynamic environments, J. Robot. Syst. **18**(12), 769–783 (2001)
42.43 A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, D. Manocha: Six degree-of-freedom haptic display of polygonal models, Proc. Conf. Vis. 2000 (2000) pp. 139–146
42.44 D.E. Johnson, P. Willemsen, E. Cohen: 6-DOF haptic rendering using spatialized normal cone search, Trans. Vis. Comput. Graph. **11**(6), 661–670 (2005)
42.45 M.A. Otaduy, M.C. Lin: A modular haptic rendering algorithm for stable and transparent 6-DOF manipulation, IEEE Trans. Vis. Comput. Graph. **22**(4), 751–762 (2006)
42.46 M.C. Lin, M.A. Otaduy: Sensation-preserving haptic rendering, IEEE Comput. Graph. Appl. **25**(4), 8–11 (2005)
42.47 T. Thompson, E. Cohen: Direct haptic rendering of complex trimmed NURBS models, Proc. ASME Dyn. Syst. Control Div. (1999)
42.48 S.P. DiMaio, S.E. Salcudean: Needle insertion modeling and simulation, IEEE Trans. Robotics Autom. **19**(5), 864–875 (2003)
42.49 B. Hannaford: Stability and performance tradeoffs in bi-lateral telemanipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1989) pp. 1764–1767

Avidbots Ex1016
Page 1110 of 2154

42.50 B. Gillespie, M. Cutkosky: Stable user-specific rendering of the virtual wall, Proc. ASME Int. Mech. Eng. Cong. Exhib., Vol. 58 (1996) pp. 397–406

42.51 R.J. Adams, B. Hannaford: Stable haptic interaction with virtual environments, IEEE Trans. Robotics Autom. **15**(3), 465–474 (1999)

42.52 B.E. Miller, J.E. Colgate, R.A. Freeman: Passive implementation for a class of static nonlinear environments in haptic display, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 2937–2942

42.53 B.E. Miller, J.E. Colgate, R.A. Freeman: Computational delay and free mode environment design for haptic display, Proc. ASME Dyn. Syst. Cont. Div. (1999)

42.54 B.E. Miller, J.E. Colgate, R.A. Freeman: Environment delay in haptic systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 2434–2439

42.55 S.E. Salcudean, T.D. Vlaar: On the emulation of stiff walls and static friction with a magnetically levitated input/output device, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 119 (1997) pp. 127–132

42.56 P. Wellman, R.D. Howe: Towards realistic vibrotactile display in virtual environments, Proc. ASME Dyn. Syst. Control Div. (1995) pp. 713–718

42.57 K. MacLean: The haptic camera: A technique for characterizing and playing back haptic properties of real environments, Proc. 5th Annu. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (1996)

42.58 A.M. Okamura, J.T. Dennerlein, M.R. Cutkosky: Reality-based models for vibration feedback in virtual environments, ASME/IEEE Trans. Mechatron. **6**(3), 245–252 (2001)

42.59 K.J. Kuchenbecker, J. Fiene, G. Niemeyer: Improving contact realism through event-based haptic feedback, IEEE Trans. Vis. Comput. Graph. **12**(2), 219–230 (2006)

42.60 D.A. Kontarinis, R.D. Howe: Tactile display of vibratory information in teleoperation and virtual environments, Presence **4**(4), 387–402 (1995)

42.61 J.T. Dennerlein, P.A. Millman, R.D. Howe: Vibrotactile feedback for industrial telemanipulators, Proc. ASME Dyn. Syst. Contr. Div., Vol. 61 (1997) pp. 189–195

42.62 A.M. Okamura, J.T. Dennerlein, R.D. Howe: Vibration feedback models for virtual environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 674–679

42.63 R.W. Lindeman, Y. Yanagida, H. Noma, K. Hosaka: Wearable vibrotactile systems for virtual contact and information display, Virtual Real. **9**(2-3), 203–213 (2006)

42.64 C. Ho, H.Z. Tan, C. Spence: Using spatial vibrotactile cues to direct visual attention in driving scenes, Transp. Res. F Traffic Psychol. Behav. **8**, 397–412 (2005)

42.65 H.Z. Tan, R. Gray, J.J. Young, R. Traylor: A haptic back display for attentional and directional cueing, Haptics-e Electron. J. Haptics Res. **3**(1), 20 (2003)

42.66 C2 Tactor: Engineering Acoustic Inc.: http://www.eaiinfo.com

42.67 W.R. Provancher, M.R. Cutkosky, K.J. Kuchenbecker, G. Niemeyer: Contact location display for haptic perception of curvature and object motion, Int. J. Robotics Res. **24**(9), 691–702 (2005)

42.68 R.S. Johansson: Sensory input and control of grip, Novartis Foundat. Symp., Vol. 218 (1998) pp. 45–59

42.69 K.O. Johnson, J.R. Phillips: A rotating drum stimulator for scanned embossed patterns and textures across the skin, J. Neurosci. Methods **22**, 221–231 (1998)

42.70 M.A. Salada, J.E. Colgate, P.M. Vishton, E. Frankel: Two experiments on the perception of slip at the fingertip, Proc. 12th Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (2004) pp. 472–476

42.71 R.J. Webster III, T.E. Murphy, L.N. Verner, A.M. Okamura: A novel two-dimensional tactile slip display: Design, kinematics and perceptual experiment, ACM Trans. Appl. Percept. **2**(2), 150–165 (2005)

42.72 N.G. Tsagarakis, T. Horne, D.G. Caldwell: SLIP AESTHEASIS: A portable 2D slip/skin stretch display for the fingertip, 1st Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (World Haptics) (2005) pp. 214–219

42.73 J. Biggs, M. Srinivasan: Tangential versus normal displacements of skin: Relative effectiveness for producing tactile sensations, Proc. 10th Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (2002) pp. 121–128

42.74 V. Hayward, J.M. Cruz-Hernandez: Tactile display device using distributed lateral skin stretch, Proc. 8th Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst. (2000) pp. 1309–1314

42.75 B. Gleeson, S. Horschel, W. Provancher: Perception of direction for applied tangential skin displacement: Effects of speed, displacement, and repetition, IEEE Trans. Haptics **3**(3), 177–188 (2010)

42.76 W.R. Provancher, N.D. Sylvester: Fingerpad skin stretch increases the perception of virtual friction, IEEE Trans. Haptics **2**(4), 212–223 (2009)

42.77 Z.F. Quek, S.B. Schorr, I. Nisky, W.R. Provancher, A.M. Okamura: Sensory substitution using 3-degree-of-freedom tangential and normal skin deformation feedback, IEEE Haptics Symp. (2014) pp. 27–33

42.78 A. Tirmizi, C. Pacchierottie, D. Prattichizzo: On the role of cutaneous force in teleoperation: Subtracting kinesthesia from complete haptic feedback, IEEE World Haptics Conf. (2013) pp. 371–376

42.79 K. Bark, J. Wheeler, P. Shull, J. Savall, M. Cutkosky: Rotational skin stretch feedback: A wearable haptic display for motion, IEEE Trans. Haptics **3**(3), 166–176 (2010)

42.80 P.B. Shull, K.L. Lurie, M.R. Cutkosky, T.F. Besier: Training multi-parameter gaits to reduce the knee adduction moment with data-driven models and haptic feedback, J. Biomech. **44**(8), 1605–1609 (2011)

42.81 K.O. Johnson, J.R. Phillips: Tactile spatial resolution. I. Two-point discrimination, gap detection,

Avidbots Ex1016
Page 1111 of 2154

grating resolution, and letter recognition, J. Neurophysiol. **46**(6), 1177–1192 (1981)
42.82 N. Asamura, T. Shinohara, Y. Tojo, N. Koshida, H. Shinoda: Necessary spatial resolution for realistic tactile feeling display, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 1851–1856
42.83 G. Moy, U. Singh, E. Tan, R.S. Fearing: Human psychophysics for teletaction system design, Haptics-e Electron. J. Haptics Res. **1**, 3 (2000)
42.84 W.J. Peine, R.D. Howe: Do humans sense finger deformation or distributed pressure to detect lumps in soft tissue, Proc. ASME Dyn. Syst. Contr. Div., Vol. 64 (1998) pp. 273–278
42.85 K.B. Shimoga: A survey of perceptual feedback issues in dexterous telemanipulation: Part II, Finger touch feedback, Proc. IEEE Virtual Real. Annu. Int. Symp. (1993) pp. 271–279
42.86 K.A. Kaczmarek, P. Bach-Y-Rita: Tactile displays. In: *Virtual Environments and Advanced Interface Design*, ed. by W. Barfield, T.A. Furness (Oxford Univ. Press, Oxford 1995) pp. 349–414
42.87 M. Shimojo: Tactile sensing and display, Trans. Inst. Electr. Eng. Jpn. E **122**, 465–468 (2002)
42.88 S. Tachi: Roles of tactile display in virtual reality, Trans. Inst. Electr. Eng. Jpn. E **122**, 461–464 (2002)
42.89 P. Kammermeier, G. Schmidt: Application-specific evaluation of tactile array displays for the human fingertip. In: *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* 2002)
42.90 S.A. Wall, S. Brewster: Sensory substitution using tactile pin arrays: Human factors, technology and applications, Signal Process. **86**(12), 3674–3695 (2006)
42.91 J.H. Killebrew, S.J. Bensmaia, J.F. Dammann, P. Denchev, S.S. Hsiao, J.C. Craig, K.O. Johnson: A dense array stimulator to generate arbitrary spatio-temporal tactile stimuli, J. Neurosci. Methods **161**(1), 62–74 (2007)
42.92 C.R. Wagner, S.J. Lederman, R.D. Howe: Design and performance of a tactile shape display using RC servomotors, Haptics-e Electron. J. Haptics Res. **3**, 4 (2004)
42.93 R.D. Howe, W.J. Peine, D.A. Kontarinis, J.S. Son: Remote palpation technology, IEEE Eng. Med. Biol. **14**(3), 318–323 (1995)
42.94 P.S. Wellman, W.J. Peine, G. Favalora, R.D. Howe: Mechanical design and control of a high-bandwidth shape memory alloy tactile display, Lect. Notes Comput. Sci. **232**, 56–66 (1998)
42.95 V. Hayward, M. Cruz-Hernandez: Tactile display device using distributed lateral skin stretch, Haptic Interfaces Virtual Environ. Teleop. Syst. Symp., Vol. 69-2 (2000) pp. 1309–1314
42.96 Q. Wang, V. Hayward: Compact, portable, modular, high-performance, distributed tactile transducer device based on lateral skin deformation, Haptic Interfaces Virtual Environ. Teleop. Syst. Symp. (2006) pp. 67–72
42.97 Q. Wang, V. Hayward: In vivo biomechanics of the fingerpad skin under local tangential traction, J. Biomech. **40**(4), 851–860 (2007)
42.98 V. Levesque, J. Pasquero, V. Hayward: Braille display by lateral skin deformation with the STReSS2 tactile transducer, 2nd Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (World Haptics) (2007) pp. 115–120
42.99 K. Drewing, M. Fritschi, R. Zopf, M.O. Ernst, M. Buss: First evaluation of a novel tactile display exerting shear force via lateral displacement, ACM Trans. Appl. Percept. **2**(2), 118–131 (2005)
42.100 K.A. Kaczmarek, J.G. Webster, P. Bach-Y-Rita, W.J. Tompkins: Electrotactile and vibrotactile displays for sensory substitution systems, IEEE Trans. Biomed. Eng. **38**, 1–16 (1991)
42.101 N. Asamura, N. Yokoyama, H. Shinoda: Selectively stimulating skin receptors for tactile display, IEEE Comput. Graph. Appl. **18**, 32–37 (1998)
42.102 L. Winfield, J. Glassmire, J.E. Colgate, M. Peshkin: T-PaD: Tactile pattern display through variable friction reduction, 2nd Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (World Haptics) (2007) pp. 421–426
42.103 J. Mullenbach, D. Johnson, J.E. Colgate, M.A. Peshkin: ActivePaD surface haptic device, IEEE Haptics Symp. (2012) pp. 407–414
42.104 O. Bau, I. Poupyrev, A. Israr, C. Harrison: TeslaTouch: Electrovibration for touch surfaces, Proc. 23nd Annu. ACM Symp. User Interface Softw. Technol. (2012) pp. 283–292
42.105 D.J. Meyer, M.A. Peshkin, J.E. Colgate: Fingertip friction modulation due to electrostatic attraction, IEEE World Haptics Conf. (2013)
42.106 H.-N. Ho, L.A. Jones: Contribution of thermal cues to material discrimination and localization, Percept. Psychophys. **68**, 118–128 (2006)
42.107 H.-N. Ho, L.A. Jones: Development and evaluation of a thermal display for material identification and discrimination, ACM Trans. Appl. Percept. **4**(2), 118–128 (2007)
42.108 D.G. Caldwell, C. Gosney: Enhanced tactile feedback (tele-taction) using a multi-functional sensory system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 955–960
42.109 D.G. Caldwell, S. Lawther, A. Wardle: Tactile perception and its application to the design of multimodal cutaneous feedback systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 3215–3221
42.110 C.G. Burdea: *Force and Touch Feedback for Virtual Reality* (Wiley, New York 1996)
42.111 M.C. Lin, M.A. Otaduy (Eds.): *Haptic Rendering: Foundations, Algorithms, and Applications* (AK Peters, Wellesley 2008)
42.112 V. Hayward, K.E. MacLean: Do it yourself haptics, Part I, IEEE Robotics Autom. Mag. **14**(4), 88–104 (2007)
42.113 K.E. MacLean, V. Hayward: Do It Yourself Haptics, Part II, IEEE Robotics Autom. Mag. **15**(1), 104–119 (2008)
42.114 V. Hayward, O.R. Astley, M. Cruz-Hernandez, D. Grant, G. Robles-De-La-Torre: Haptic interfaces and devices, Sensor Rev. **24**(1), 16–29 (2004)

Avidbots Ex1016
Page 1112 of 2154

42.115 K. Salisbury, F. Conti, F. Barbagli: Haptic rendering: Introductory concepts, IEEE Comput. Graph. Appl. **24**(2), 24–32 (2004)
42.116 V. Hayward, K.E. MacLean: A brief taxonomy of tactile illusions and demonstrations that can be done in a hardware store, Brain Res. Bull. **75**(6), 742–752 (2007)
42.117 G. Robles-De-La-Torre: The importance of the sense of touch in virtual and real environments, IEEE Multimedia **13**(3), 24–30 (2006)
42.118 Haptics-e: The Electronic Journal of Haptics Research, http://www.haptics-e.org
42.119 Haptics Technical Committee: http://www.worldhaptics.org

Avidbots Ex1016
Page 1113 of 2154

Multimedia Contents

43. Telerobotics

Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, Dongjun Lee

In this chapter we present an overview of the field of telerobotics with a focus on control aspects. To acknowledge some of the earliest contributions and motivations the field has provided to robotics in general, we begin with a brief historical perspective and discuss some of the challenging applications. Then, after introducing and classifying the various system architectures and control strategies, we emphasize bilateral control and force feedback. This particular area has seen intense research work in the pursuit of telepresence. We also examine some of the emerging efforts, extending telerobotic concepts to unconventional systems and applications. Finally, we suggest some further reading for a closer engagement with the field.

43.1 **Overview and Terminology** 1085
43.2 **Telerobotic Systems and Applications** 1087
43.2.1 Historical Perspective 1087
43.2.2 Applications 1090
43.3 **Control Architectures** 1090
43.3.1 Supervisory Control 1091
43.3.2 Shared Control 1092
43.3.3 Direct and Bilateral Teleoperation 1093
43.4 **Bilateral Control and Force Feedback** 1095
43.4.1 Position/Force Control 1095
43.4.2 Passivity and Stability 1096
43.4.3 Transparency and Multichannel Feedback 1097
43.4.4 Time Delay and Scattering Theory 1098
43.4.5 Wave Variables 1098
43.4.6 Teleoperation with Lossy Communication 1099
43.4.7 Port-Based Approaches 1099
43.5 **Emerging Applications of Telerobotics** ... 1101
43.5.1 Telerobotics for Mobile Robots 1101
43.5.2 Multilateral Telerobotics 1102
43.6 **Conclusions and Further Reading** 1104
Video-References 1104
References 1105

43.1 Overview and Terminology

Telerobotics is perhaps one of the earliest aspects and manifestations of robotics. Literally meaning robotics at a distance, it is generally understood to refer to robotics with a human operator in control or *human-in-the-loop*. Any high-level, planning, or cognitive decisions are made by the human user, while the robot is responsible for their mechanical implementation. In essence, the *brain* is removed or distant from the *body*.

Herein the term *tele*, which is derived from the Greek and means distant, is generalized to imply a barrier between the user and the environment. This barrier is overcome by remote-controlling a robot at the environment, as indicated in Fig. 43.1. Besides distance, barriers may be imposed by hazardous environments or scaling to very large or small environments. All barriers prevent the user from physically reaching or directly interacting with the environment.

While the physical separation may be very small, with the human operator and the robot sometimes occupying the same room, telerobotic systems are often at least conceptually split into two sites. The *local site* encompasses the human operator and all elements necessary to support the system's connection with the user, which could be joysticks, monitors, keyboards, or other

Avidbots Ex1016
Page 1114 of 2154

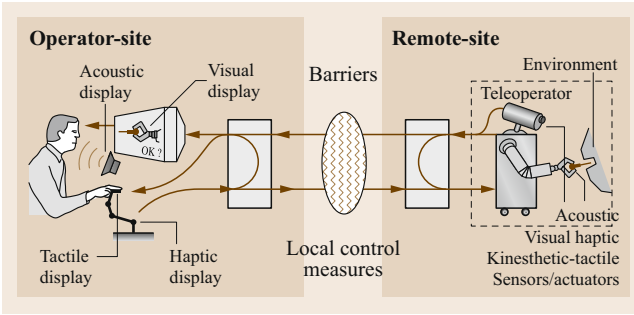


Fig. 43.1 Overview of a telerobotic system (after [43.1], adapted from [43.2])

input/output devices. The *remote site* contains the robot, supporting sensors and control elements, and the environment to be manipulated.

To support its operation, telerobotics integrates many areas of robotics. At the remote site, to operate the robot and execute the human's commands, the system may control the motion and/or forces of the robot. We refer to Chaps. 7 and 8 for detailed descriptions of these areas. Also, sensors are invaluable (Chap. 5), including force sensors (Chap. 28) and others (Part C). Meanwhile, at the local site information is often displayed haptically (Chap. 41).

A recent addition to telerobotics is the use of computer networks to transmit information between the sites. The ubiquity of network access is allowing remote control from anywhere on demand. Chapter 44 discusses some of these developments, detailing network infrastructure and focusing on visual, often web-based, user interfaces, avoiding the need for specialized mechanical I/O (input/output) devices. Computer networks also allow new multilateral telerobotic architectures. For example, multiple users may share a single robot or a single user may control multiple robots (Sect. 43.5.2). Unfortunately, computer networks often see transmission delays and can introduce nondeterministic effects such as variable delay times and data losses. These effects can easily destabilize force feedback loops and require particular countermeasures (Sects. 43.4.4–43.4.6).

We should also point out the relation between telerobotics and human exoskeletons, as described in Chap. 69. Exoskeletons are also controlled by a human operator, leaving all planning and high-level challenges to the user, and their control systems share many aspects with telerobotics. However, the two sites are physically combined in an exoskeleton as the user directly touches and interacts with the robot. In this chapter, we will disallow any such direct mechanical connection.

The inclusion of the human operator makes telerobotics very attractive to handle unknown and unstructured environments. Applications are plentiful (Part F) and range from space robotics (Chap. 55) to dealing with hazardous environments (Chap. 58), from search and rescue situations (Chap. 60), to medical systems (Chap. 63) and rehabilitation (Chap. 64).

Before proceeding, we define some basic terminology. Indeed many other terms are used nearly synonymously with telerobotics, in particular *teleoperation* and *telemanipulation*. Telerobotics is the most common, emphasizing a human's (remote) control of a robot. Teleoperation stresses the task-level operations, while telemanipulation highlights object-level manipulation.

Within telerobotics, a spectrum of control architectures is used. *Direct control* or *manual control* falls at one extreme, indicating that the user is controlling the motion of the robot directly and without any automated help. At the other extreme, *supervisory control* implies that user's commands and feedback occur at a very high level and the robot requires substantial intelligence or autonomy to fulfill its function. Between the two extrema lie a variety of *shared control* architectures, where some degree of autonomy or automated help is available to assist the user.

In practice, many systems involve at least some level of direct control and accept the user's motion commands via a joystick or similar device in the user interface. The joystick is an instrumented mechanism and can itself be viewed as a robot. The local and remote robots are called *master* and *slave* respectively, while the system is referred to as a *master–slave system*. To provide direct control, the slave robot is programmed to follow the motions of the master robot, which is positioned by the user. It is not uncommon for the master robot (joystick) to be a kinematic replica of the slave, providing an intuitive interface.

Some master–slave systems provide force feedback, such that the master robot not only measures motions but also displays forces to the user. The user interface becomes fully bidirectional and such telerobotic systems are often called *bilateral*. The human–master interactions are a form of human–robot interaction (Chap. 69). The field of haptics (Chap. 41) also discusses bidirectional user interfaces, involving both motion and force, though more commonly to interface the user with virtual instead of remote environments. We should note that both motion and force may become the input or output to/from the user, depending on the system architecture.

Finally, *telepresence* is often discussed as an ultimate goal of master–slave systems and telerobotics in general. It promises to the user not only the abil-

Avidbots Ex1016
Page 1115 of 2154

ity to manipulate the remote environment, but also to perceive the environment as if encountered directly. The human operator is provided with enough feedback and sensations to feel present in the remote site. This combines the haptic modality with other modalities serving the human senses of vision, hearing or even smell and taste. See videos VIDEO 297, VIDEO 318 and VIDEO 319, and VIDEO 321 for some early and recent results aiming for this telepresence. We focus our descriptions on the haptic channel, which is created by the robotic hardware and its control systems. The master–slave system becomes the medium through which the user interacts with the remote environment and ideally they are fooled into forgetting about the medium itself. If this is achieved, we say that the master–slave system is transparent.

While bilateral master–slave systems have held the biggest promise for telepresence and intuitive operations, they have also posed some of the largest stability and control problems. Especially considering force feedback from sensors at the remote site, these systems close multiple interwoven feedback loops and have to deal with large uncertainties in the environment. They have received heavy research attention and will therefore be a repeated focus in some of our following discussions.

43.2 Telerobotic Systems and Applications

Telerobotic systems, like most robotic devices, are typically designed for specific tasks and according to explicit requirements. As such, many unique systems have evolved, of which we present an overview for different applications. We begin with a short historical perspective, then describe different applications with various robot designs and user interfaces.

43.2.1 Historical Perspective

Teleoperation enjoys a rich history and dates back to nuclear research by Raymond C. Goertz in the 1940s and 1950s. In particular, he created systems for humans to handle radioactive material from behind shielded walls. The first systems were electrical, controlled by an array of on–off switches to activate various motors and move various axes [43.3]. Without any feel, these manipulators were *slow and somewhat awkward to operate*, leading *Goertz* to build pairs of mechanically linked master–slave robots [43.3, 4]. Connected by gears, linkages, and cables, these systems allowed the operator to use natural hand motions and transmitted forces and vibrations through the connecting structure (Fig. 43.2). Unfortunately they limited the distance between the operator and environment and required the use of kinematically identical devices. *Goertz* quickly recognized the value of electrically coupled manipulators and laid the foundations of modern telerobotics and bilateral force-reflecting positional servos [43.5].

At the beginning of the 1960s the effects of time delay on teleoperation started to become a topic of research [43.6, 7]. To cope with this problem the concept of supervisory control was introduced [43.2] and inspired the next years of development. In the late 1980s and early 1990s theoretical control came into play with Lyapunov-based analysis and network theory [43.8–13]. Using these new methods, bilateral control of telerobotic systems became the vital research area it is today (Sect. 43.4). The growth of the Internet and its use as a communication medium has further fueled this trend, adding the challenges of nondeterministic time delay.

On the hardware side, the Central Research Laboratory model M2 of 1982 was the first telerobotic system which realized force feedback while separating master and slave electronics. It was developed together with the Oak Ridge National Laboratory and was used for some time for a wide range of demonstration tasks including military, space or nuclear applications. The National Aeronautics and Space Administration (NASA) tested the M2 system to simulate the ACCESS space truss assembly with excellent results (Fig. 43.3). The ad-

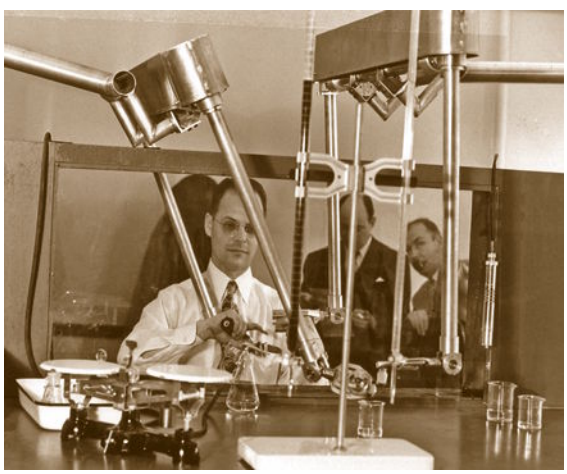


Fig. 43.2 Raymond C. Goertz used electrical and mechanical teleoperators in the early 1950s to handle radioactive material (courtesy Argonne National Labs)

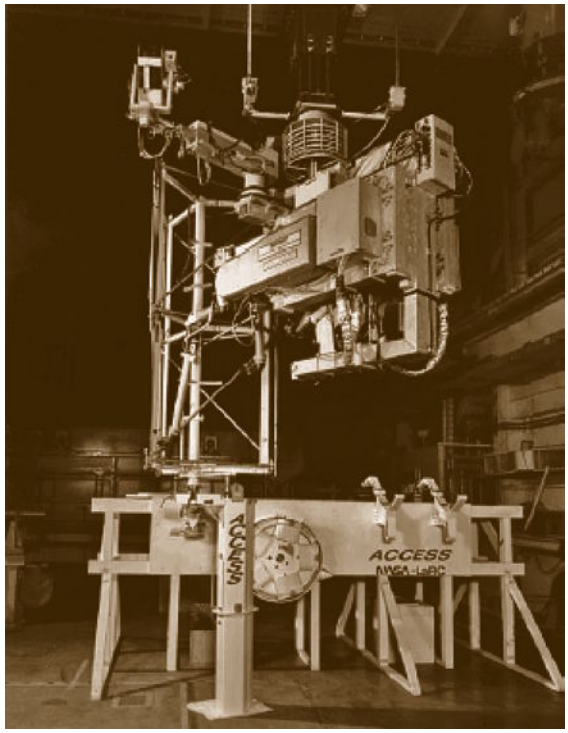


Fig. 43.3 The telerobotic system CRL Model M2 is used to verify the assembly of space truss structures (1982) (courtesy Oak Ridge National Laboratory)

vanced servomanipulator (ASM) was developed from the M2 to improve the remote maintainability of manipulators and intended as a foundation for telerobotic systems [43.14].

Also driven by the nuclear application, bilateral servomanipulators for teleoperation were developed in France at the Commission de Energie Atomique (CEA) by *Vertut* and *Coiffet* [43.15]. With the MA 23 they demonstrated telerobotic operation including computer-assisted functionalities to improve the operator's performance [43.16]. The assistance included software jigs and fixtures or virtual walls and restrictions [43.17] (Sect. 43.3.2).

For space applications a dual-arm force reflecting telerobotic system was developed by *Bejczy* at the Jet Propulsion Laboratory (JPL) [43.18] and VIDEO 298. This system was the first use of kinematically and dynamically different master and slave robots. It required control in the Cartesian space coordinates of the operator's hand and slave robot's tool. Figure 43.4 shows the master control station with its two back-drivable hand controllers. The system was used to simulate teleoperation in space.

In 1993 the first telerobotic system was flown in space with the German Spacelab Mission D2 on board



Fig. 43.4 JPL ATOP control station (early 1980s) (JPL No. 19902Ac, courtesy NASA/JPL-CALTECH)

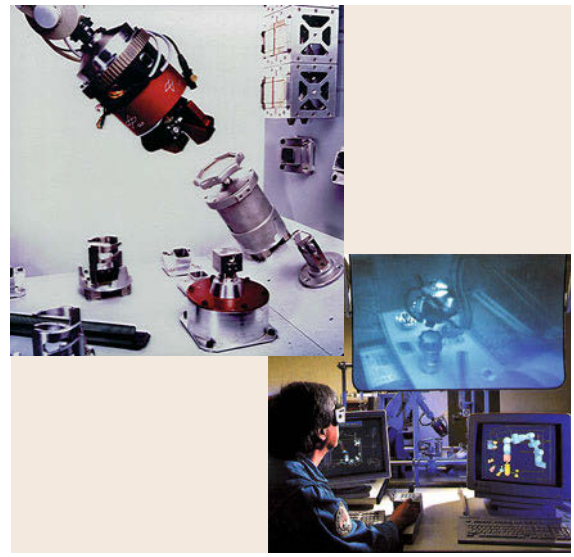


Fig. 43.5 ROTEX, the first remotely controlled robot in space (1993). Telerobot in space and ground operator station (courtesy German Aerospace Center, DLR)

the Space Shuttle Columbia. The robot technology experiment (ROTEX), shown in Fig. 43.5, demonstrated remote control of a space robot by means of local sensory feedback, predictive displays, and teleoperation [43.19]. In this experiment the round trip delay was 6–7 s, such that it was not feasible to include force feedback into the control loop.

Throughout the 1980s and 1990s, as nuclear power activities began to decline, interests expanded into new areas including medicine and undersea operations. Ef-

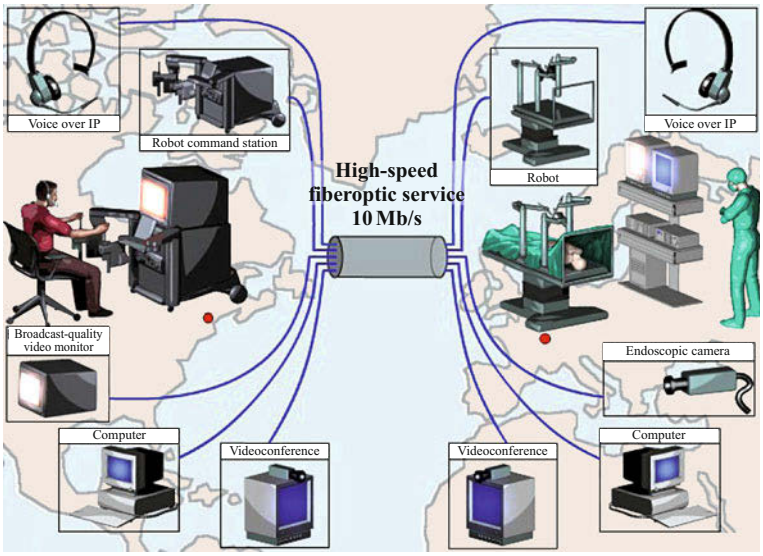


Fig. 43.6 Operation Lindberg. The first transcontinental telerobotic surgery (2001) (courtesy M. Ghodoussi)

forts were accelerated by the availability of increasing computer power as well as the introduction of novel hand controllers, e.g., the PHANToM device [43.20], popularized by haptic applications in virtual reality (Chap. 41).

Simultaneously, surgery was seeing the trend toward minimally invasive techniques, highlighted by the first laparoscopic cholecystectomy (removal of the gallbladder) in 1987. Several groups saw the potential for telerobotics and pursued telesurgical systems. Most noteworthy are the Telepresence Surgery System developed at the Stanford Research Institute (now SRI International) in 1987 [43.21], the Laparoscopic Assistant Robotic System (LARS) created at the IBM Watson Research Center [43.22], the teleoperated surgical instrument Falcon designed at MIT (Massachusetts Institute of Technology) [43.23], and the Robot Assisted Microsurgery (RAMS) workstation developed at JPL [43.24].

In 1995 Intuitive Surgical Inc. was founded to leverage several of these concepts, leading to the da Vinci telesurgical system [43.25] and its introduction to market in 1999. Meanwhile Computer Motion started with a voice-controlled robot moving an endoscopic camera [43.26] and extended those capabilities into the



Fig. 43.7 Intuitive Surgical Inc. makes the da Vinci telerobotic system, which is used in minimally invasive surgery (courtesy 2008 Intuitive Surgical, Inc.)



Fig. 43.8 tEODor, a telerobotic system for disarming of explosives (courtesy telerob Gesellschaft für Fernhantierungstechnik mbH, Ostfildern, Germany)

Avidbots Ex1016
Page 1118 of 2154

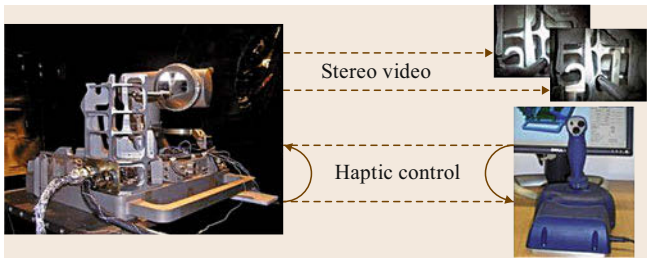


Fig. 43.9 ROKVISS, a telerobotic system providing stereo vision and haptic feedback to the ground operator (courtesy German Aerospace Center, DLR)

ZEUS system. In 2001 a surgeon in New York (USA) used a ZEUS system to perform the first transatlantic telesurgical laparoscopic cholecystectomy on a patient located in Strasbourg (France) [43.27], as depicted in Fig. 43.6. The system did not include force feedback, so the surgeon had to rely on visual feedback only.

In this perspective we have given reference only to the systems that may be seen as milestones within the history of telerobotics. Other systems, which have been developed and added value to the research field, unfortunately could not be mentioned here.

43.2.2 Applications

Telerobotic systems have been motivated by issues of human safety in hazardous environments (e.g., nuclear or chemical plants), the high cost of reaching remote environments (e.g., space), scale (e.g., power amplification or position scaling in micromanipulation or minimally invasive surgery), and many others. Not surprisingly, after their beginning in nuclear research, telerobotic systems have evolved to many fields of application. Nearly everywhere a robot is used, telerobotic systems can be found. The following are some of the more exciting uses.

In minimally invasive surgery telerobots allow procedures to be performed through small incisions, reducing the trauma to the patient compared to traditional surgery [43.28]. The da Vinci system, made by Intuitive Surgical Inc. [43.25] and shown in Fig. 43.7, is the only commercially available device at present. Other efforts, however, have included computer motion [43.26] and endoVia Medical [43.29] on the commercial side, as well as the University of Washington [43.30], Johns Hopkins University [43.31], the German Aerospace Center [43.32], and many others (VIDEO 322).

Protecting the operator from having to reach into a hazardous environment, telerobotic systems are widely used in nuclear or chemical industry. Some systems have been developed for the maintenance of high-voltage electrical power lines, which can be safely repaired without service interruption by a human operator using a telerobotic system. Disarming of explosives is another important task. Many systems like the telerob explosive ordnance disposal and observation robot (tEODor) shown in Fig. 43.8 or PackBot, made by iRobot [43.33], are used by police and military to disarm mines and other explosives. Similar vehicles are remote controlled for search and rescue in disaster zones [43.34].

Space robotics is a classic application, in which distance is the dominating barrier, as discussed in Chap. 55. The NASA rovers on Mars are a famous example. Due to the time delay of several minutes, the rovers are commanded using supervisory control, in which the human operator is defining the goal of a movement and the rover achieves the goal by local autonomy using sensory feedback directly [43.35].

In orbital robotics the German technology experiment ROKVISS (robot component verification on the international space station (ISS)) was the most advanced telerobotic system [43.36]. Launched in 2004 and operational through 2010, it was installed outside the Russian module of the international space station. It validated advanced robot components in the slave system, including torque sensors and stereo video cameras, in real space conditions. Using a direct communication link between the space station and the operator station at DLR (German Aerospace Center), the time delay was kept at about 20 ms allowing a bilateral control architecture with high-fidelity force feedback to the operator [43.37] (Fig. 43.9). This technology is leading toward robotic service satellites, called Robonauts, which can be controlled remotely from the ground to help real astronauts during extravehicular activities (EVA) or to perform repair and maintenance tasks [43.38].

43.3 Control Architectures

Compared to plain robotic systems, in which a robot executes a motion or other program without further consultation of a user or operator, telerobotic systems provide information to and require commands from the user. Their control architectures can be described by the style and level of this connection, as shown in Fig. 43.10. Organized in a spectrum, the three main categories are:

- Direct control
- Shared control
- Supervisory control.

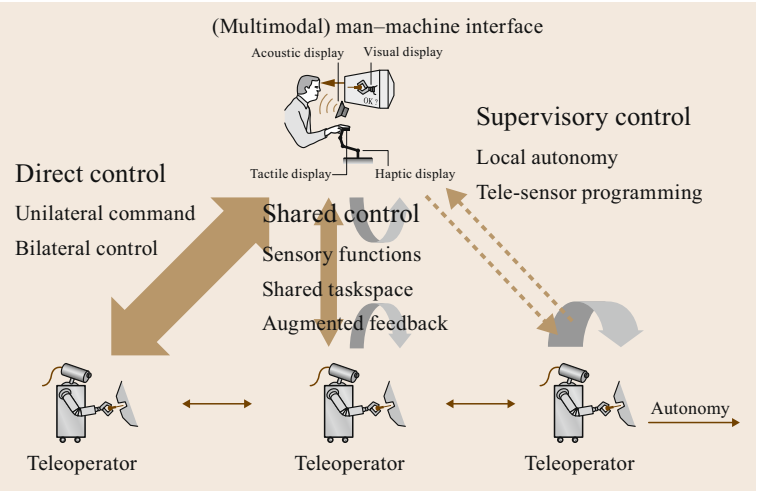


Fig. 43.10 Different concepts for telerobotic control architectures

In practice, however, control architectures often include parts of all strategies.

Direct control implies no intelligence or autonomy in the system, so that all slave motion is directly controlled by the user via the master interface. This may incorporate sensory feedback to the user in a bilateral configuration. If the slave motion is controlled by a combination of direct user commands and local sensory feedback or autonomy, the architecture is denoted as shared control. It is similarly shared if user feedback is augmented from virtual reality or by other automatic aids. In supervisory control user commands and feedback occur at a higher level. The connection is more loose and the slave has to rely on stronger local autonomy to refine and execute tasks. The following explains the architectures in reverse order, leading to a detailed treatment of direct and bilateral control in Sect. 43.3.3, which introduces the basic ideas for Sect. 43.4.

43.3.1 Supervisory Control

Supervisory control, introduced by *Ferell* and *Sheridan* in 1967 [43.2], is derived from the analog of supervising a human subordinate staff member. The supervisor gives high-level directives to and receives summary information from, in this case, the robot. *Sheridan* describes this approach in comparison with manual and automatic robot control [43.39]:

> *Human operators are intermittently programming and continually receiving information from a computer that itself closes an autonomous control loop through artificial effectors and sensors.*

In general, supervisory control techniques will allow more and more autonomy and intelligence to shift to the robot system. Today simple autonomous control loops may be closed at the remote site, with only state and model information being transmitted to the operator site. The operator supervises the telerobotic system closely and decides exactly how to act and what to do. A specific implementation of supervisory control is the telesensor programming approach, which is presented hereafter. See also VIDEO 299 for another implementation of supervisory control for space operation.

Telesensor Programming

Developed for space applications with large communication delays, the telesensor programming (TSP) approach has been characterized as a *task-level-oriented programming technique* and *sensor-based teaching by showing* [43.40, 41]. In essence, operators interact with a complex simulation of the robot and remote environment, in which they can test and adjust tasks. The tasks, consisting of robot and environment signals and configuration parameters, are then uploaded to the remote site. The approach presumes that the sensor systems provide sufficient information about the actual environment so that the tasks can be executed autonomously. Specifications and high-level planning remain the responsibility of the human operator.

Figure 43.11 shows the structure of a TSP implementation, consisting of two control loops working in parallel. One loop controls the real (remote) system, which contains internal feedback for local autonomy. The other loop establishes a simulation environment which is structurally equivalent to the real system, with a few exceptions. Most importantly, any signal delay which may result from communication to the remote system, in particular in space applications, is not duplicated in the simulation. This makes the simulation predictive with respect to the real system. A second exception is the display of internal variables in the sim-

Avidbots Ex1016
Page 1120 of 2154

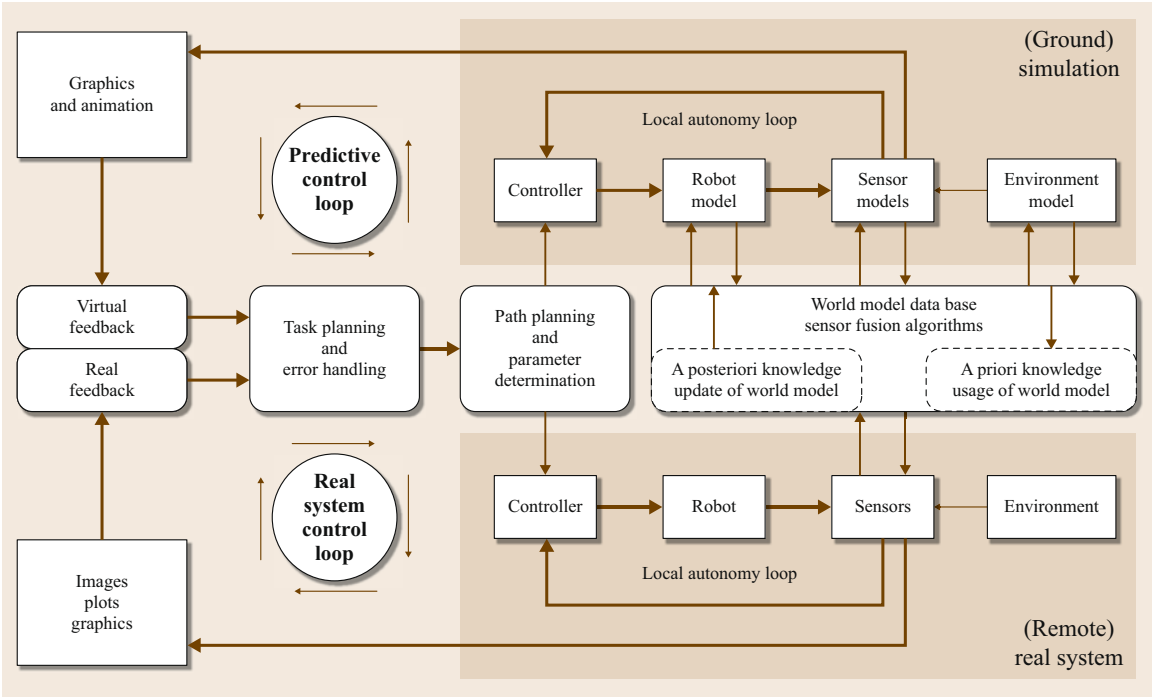


Fig. 43.11 The concept of telesensor programming as demonstrated during the ROTEX mission

ulation, which cannot be observed (measured) in the real system. This gives the operator or task planner more insight into what is happening or may happen in the system in response to commands. Communication between the two loops occurs via a common model data base which delivers a priori knowledge for execution on the remote system and a posteriori knowledge for model updating in the simulated world.

Unique tools are necessary to implement the functionality required for such a telerobotic control system. First a sophisticated simulation system has to be provided to emulate the real robot system. This includes the simulation of sensory perception within the real environment. Also, the operator needs an efficient interface to set up task descriptions, to configure the task control parameters, to decide what kind of sensors and control algorithms should be used, and to debug an entire job execution phase.

For telerobotic systems with large time delays of a few seconds or more, e.g., in space and undersea applications, such a sensor-based task-directed programming approach has advantages. It is not feasible for human operators to handle the robot movements directly under delayed visual feedback. Only a predictive simulation allows the operator to telemanipulate the remote system [43.42]. In addition, the use of force reflecting hand controllers to feed back force signals from the simulated predicted world can improve the operator's performance [43.43]. Finally, an interactive supervisory user interface makes it possible to configure the environmental and control parameters.

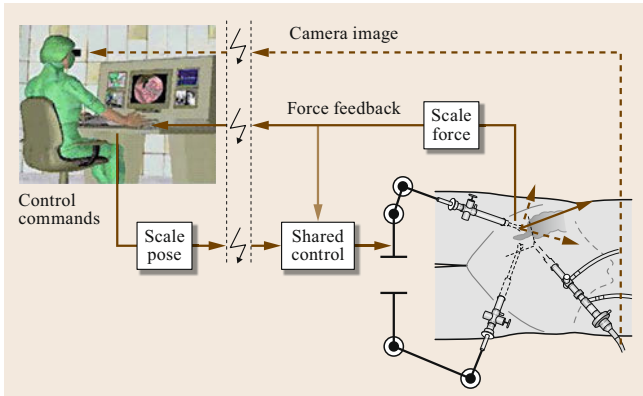


Fig. 43.12 An example for the shared control concept in telerobotic surgery

43.3.2 Shared Control

Shared control tries to combine the basic reliability and sense of presence achievable by direct control with the smarts and possible safety guarantees of autonomous control ([43.44, 45] and VIDEO 299). This may occur in various forms. For example, the slave robot may need

to correct motion commands, regulate subsets of joints or subtasks, or overlay additional commands.

With large communication delays, a human operator may only be able to specify gross path commands, which the slave must fine-tune with local sensory information [43.46]. We may also want the slave to assume control of subtasks, such as maintaining a grasp over long periods of time [43.47]. And in surgical applications, shared control has been proposed to compensate for beating heart movements (Fig. 43.12). The sensed heart motion is overlaid on the user commands, so the surgeon can operate on a virtually stabilized patient [43.48].

A special application of shared control is the use of *virtual fixtures* ([43.49–51] and VIDEO 72). Virtual elements, such as virtual surfaces, virtual velocity field, guide tubes, or other appropriate objects, are superimposed into the visual and/or haptic scene for the user. These fixtures can help the operator perform tasks by limiting movement into restricted regions and/or influencing movement along desired paths. Control is thus shared at the master site, taking advantage of preknowledge of the system or task to modify the user's commands and/or to combine them with autonomously generated signals.

Capitalizing on the accuracy of robotic systems while sharing control with the operator, telerobotic systems with virtual fixtures can achieve safer, faster and more intuitive operation. *Abbott* et al. describe the benefits by comparison to the common physical fixture of a ruler [43.50]:

> *A straight line drawn by a human with the help of a ruler is drawn faster and straighter than a line drawn freehand. Similarly, a [master] robot can apply forces or positions to a human operator to help him or her draw a straight line.*

Based on the nature of the master robot and its controller, the virtual fixtures may apply corrective forces or constrain positions. In both cases, and in contrast to physical fixtures, the level and type of assistance can be programmed and varied.

43.3.3 Direct and Bilateral Teleoperation

To avoid difficulties in creating local autonomy, most telerobotic systems include some form of direct control: they allow the operator to specify the robot's motions. This may involve commanding either position or velocity or acceleration. We begin our discussions with the later two options, which are generally implemented unilaterally without force feedback to the user. We then focus on position control, which is more suited to bilateral operation. We will assume a master–slave system, i. e., the user is holding a joystick or master mechanism serving as an input device.

Unilateral Acceleration or Rate Control

For underwater, airborne, or space applications, a slave robot may be a vehicle actuated by thrusters. Direct control thus requires the user to power the thrusters, which in turn accelerates the vehicle. For other applications, the user may be required to command the rate or velocity of the vehicle or slave robot. In both scenarios, the input device is commonly a joystick, often spring centered, where the acceleration or rate commands are proportional to the joystick displacement. For six degree-of-freedom (DOF) applications, i. e., when the slave needs to be controlled in translation and orientation, a six-dimensional (6-D) space mouse can be used. Alternatively two joysticks may separately command translation and orientation.

Acceleration and rate control are very attractive when the master and slave robots are fundamentally different, for example if the slave robot can reach an effectively unbounded workspace. Unfortunately, basic implementations can require considerable effort for the operator to reach and hold a given target location. As expected, users can more accurately position a system under rate control than under acceleration control [43.52]. Indeed acceleration control necessitates users to regulate a second-order system versus a first-order system for rate control. Assuming the slave has local position feedback available, a control system is often incorporated locally, such that the user may specify position commands and is relieved from the dynamic control problem. We refer to Sect. 43.5.1 for some emerging developments for bilateral control of mobile robots.

Position Control and Kinematic Coupling

Assuming that the slave is under position control, we can consider a kinematic coupling between master and slave, i. e., a mapping between master and slave positions. In particular, we must remember that the master mechanism moves in the master workspace, while the slave robot moves in the slave workspace. The mapping connects these two spaces, which are nearly always somewhat different.

Clutching and Offsets. Before discussing how the two robots are coupled, we must understand that they are not always coupled. For example, before the system is turned on, master and slave robots may, for whatever reason, be placed in some initial position/configuration. We have three options of how to engage the system:

1. First autonomously move one or both robots so they come to the same position

Avidbots Ex1016
Page 1122 of 2154

2. Wait until someone (the user) externally moves one robot to match the location of the other, or
3. Connect the two robots with some offset.

Once connected, most systems also allow a temporary disconnection between the two sites. The reason is twofold: to allow the user to rest without affecting the slave state and to allow a shift between the two robots. The later is most important if the workspaces of both robots do not perfectly overlap. This is much like picking up your mouse off your mouse pad to reposition without moving the cursor. In telerobotics the process is called *clutching* or sometimes *indexing*. If clutching is allowed, or both robots are not constrained to start at the same location, the system must allow for offsets between the two robots.

When clutched or disconnected, most systems hold the slave at rest or allow it to float in response to environment forces. It is also possible for the slave to retain its preclutching momentum and continue moving, similar to kinetic scrolling popularized in smartphones [43.53].

Kinematically Similar Mechanisms. The simplest scenario involves a master and slave mechanism that are kinematically equivalent if not entirely identical. In this case, the two robots can be connected at a joint level. With $\boldsymbol{q}$ denoting joint values and subscripts "m" referring to the master, "s" to the slave, "offset" to a shared offset, and "d" to a desired value, we can write

$$\begin{aligned} \boldsymbol{q}_{\mathrm{sd}} &= \boldsymbol{q}_{\mathrm{m}} + \boldsymbol{q}_{\mathrm{offset}} \,, \\ \boldsymbol{q}_{\mathrm{md}} &= \boldsymbol{q}_{\mathrm{s}} - \boldsymbol{q}_{\mathrm{offset}} \,. \end{aligned} \tag{43.1}$$

At the instance the two robots are to be connected or reconnected, the offset is computed as

$$\boldsymbol{q}_{\mathrm{offset}} = \boldsymbol{q}_{\mathrm{s}} - \boldsymbol{q}_{\mathrm{m}} \,. \tag{43.2}$$

Most kinematically similar master–slave systems have the same workspace at both sites and do not allow clutching. By construction the offset is then always zero.

Depending on the controller architecture, the joint velocities may be similarly related, taking derivatives of (43.1). An offset in velocities is not necessary.

Kinematically Dissimilar Mechanisms. In many cases, the master and slave robots differ. Consider that the master is connected to the human user and thus should be designed accordingly. Meanwhile the slave works in some environment and may have a very different joint configuration and different number of joints. As a result, connecting the robots joint by joint may not be feasible or appropriate.

Instead kinematically dissimilar robots are commonly connected at their tips. If $\boldsymbol{x}$ is a robot's tip position, we have

$$\begin{aligned} \boldsymbol{x}_{\mathrm{sd}} &= \boldsymbol{x}_{\mathrm{m}} + \boldsymbol{x}_{\mathrm{offset}} \,, \\ \boldsymbol{x}_{\mathrm{md}} &= \boldsymbol{x}_{\mathrm{s}} - \boldsymbol{x}_{\mathrm{offset}} \,. \end{aligned} \tag{43.3}$$

If orientations are also connected, with $\mathbf{R}$ describing a rotation matrix, we have

$$\begin{aligned} \mathbf{R}_{\mathrm{sd}} &= \mathbf{R}_{\mathrm{m}}\mathbf{R}_{\mathrm{offset}} \,, \\ \mathbf{R}_{\mathrm{md}} &= \mathbf{R}_{\mathrm{s}}\mathbf{R}_{\mathrm{offset}}^{\mathrm{T}} \,, \end{aligned} \tag{43.4}$$

where the orientational offset is defined as slave relative to master

$$\mathbf{R}_{\mathrm{offset}} = \mathbf{R}_{\mathrm{m}}^{\mathrm{T}}\mathbf{R}_{\mathrm{s}} \,. \tag{43.5}$$

Again velocities and angular velocities may be connected if needed and do not require offsets.

Finally note that most telerobotic systems use a video camera at the remote site and a monitor at the local site. To make the connection appear natural, the slave position and orientation should be measured relative to the camera, while the master position and orientation should be measured relative to the user's view.

Scaling and Workspace Mapping. Kinematically dissimilar master–slave robots are commonly also of different size. This means not only do they require clutching to fully map one workspace to another, but they often necessitate motion scaling. And so (43.3) becomes

$$\begin{aligned} \boldsymbol{x}_{\mathrm{sd}} &= \mu\boldsymbol{x}_{\mathrm{m}} + \boldsymbol{x}_{\mathrm{offset}} \,, \\ \boldsymbol{x}_{\mathrm{md}} &= \frac{(\boldsymbol{x}_{\mathrm{s}} - \boldsymbol{x}_{\mathrm{offset}})}{\mu} \,. \end{aligned} \tag{43.6}$$

The orientation, however, typically should not be scaled. The scale μ may be set to either map the two workspaces as best possible, or to provide the most comfort to the user.

If force feedback is provided, as described below, an equivalent force scale may be desired. This will prevent distortion of the remote environmental conditions, such as stiffness or damping, by the scaling. In addition to the motion and force scalings, it is also possible to directly achieve power scaling between master and slave systems [43.51].

Beyond linear scaling, several research efforts have created nonlinear or time-varying mappings, which deform the workspaces. These may effectively change the scale in the proximity of objects [43.54] or drift the offset to best utilize the master workspace [43.55].

Avidbots Ex1016
Page 1123 of 2154

Local Position and Advanced Control. By construction we are now assuming that the slave follows a position command. This necessitates a local slave controller to regulate its position. In particular for kinematically dissimilar mechanisms, this will be a Cartesian tip position controller (Chap. 7).

If the slave robot has redundancies or possesses a large number of DOFs, these may be controlled either automatically to optimize some criterion or manually with additional user commands. Indeed some emerging applications are coordinating multiple users to control such complex systems. This is particularly relevant when the kinematic dissimilarity becomes extreme and has received considerable research attention. We refer here to Chap. 11 and especially Sect. 43.5 for appropriate techniques and new developments.

43.4 Bilateral Control and Force Feedback

In pursuit of telepresence and to increase task performance, many master–slave systems incorporate force feedback. That is, the slave robot doubles as a sensor and the master functions as a display device, so that the system provides both forward and feedback pathways from the user to the environment and back. Figure 43.13 depicts the common architecture viewed as a chain of elements from the user to the environment.

The bilateral nature of this setup makes the control architecture particularly challenging: multiple feedback loops form and even without environment contact or user intervention, the two robots form an internal closed loop. The communications between the two sites often inserts delays into the system and this loop, so that stability of the system can be a challenging issue [43.56].

To present force information without stability problems, it is possible to use alternate displays, such as audio or tactile devices [43.57]. Meanwhile, the combination of vibrotactile methods with explicit force feedback can increase high-frequency sensations and provide benefits to the user [43.58]. Tactile shape sensing and display also extends the force information presented to the user [43.59].

In the following we discuss explicit force feedback. We first examine the basic architectures before discussing stability and some advanced techniques.

43.4.1 Position/Force Control

Two basic architectures couple the master and slave robots: position–position and position–force. We assume that the robot tips are to be connected by the equations of Sect. 43.3.3 and give the control laws for translation. Control of orientation or joint motions follows equivalent patterns.

Position–Position Architecture

In the simplest case, both robots are instructed to track each other. Both sites implement a tracking controller, often a proportional–derivative (PD) controller, to fulfill these commands,

$$\begin{aligned} \boldsymbol{F}_\mathrm{m} &= -\boldsymbol{K}_\mathrm{m}(\boldsymbol{x}_\mathrm{m} - \boldsymbol{x}_\mathrm{md}) - \boldsymbol{B}_\mathrm{m}(\dot{\boldsymbol{x}}_\mathrm{m} - \dot{\boldsymbol{x}}_\mathrm{md}) \,, \\ \boldsymbol{F}_\mathrm{s} &= -\boldsymbol{K}_\mathrm{s}(\boldsymbol{x}_\mathrm{s} - \boldsymbol{x}_\mathrm{sd}) - \boldsymbol{B}_\mathrm{s}(\dot{\boldsymbol{x}}_\mathrm{s} - \dot{\boldsymbol{x}}_\mathrm{sd}) \,. \end{aligned} \tag{43.7}$$

If the position and velocity gains are the same ($\boldsymbol{K}_\mathrm{m} = \boldsymbol{K}_\mathrm{s} = \boldsymbol{K}$, $\boldsymbol{B}_\mathrm{m} = \boldsymbol{B}_\mathrm{s} = \boldsymbol{B}$), then the two forces are the same and the system effectively provides force feedback. This may also be interpreted as a spring and damper between the tips of each robot, as illustrated in Fig. 43.14. If the two robots are substantially different and require different position and velocity gains, some suitable force, position or power scalings, as explained in Sect. 43.3.3, may be utilized.

Note we have assumed the slave is under impedance control and back-drivable. If the slave is admittance controlled, i. e., it accepts position commands directly, the second part of (43.7) is unnecessary.

Also note that by construction the user feels the slave's controller forces, which include forces associated with the spring–damper and slave inertia in addition to environment forces. Indeed while moving without contact, the user will feel the inertial and other dynamic forces needed to move the slave. Furthermore, if the slave is not back-drivable, i. e., does not easily move under environment forces, the environment force

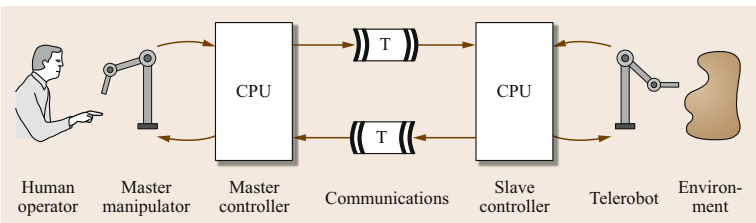


Fig. 43.13 A typical bilateral teleoperator can beviewed as a chain of elements reaching from user to environment (CPU – central processing unit)

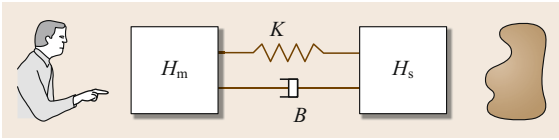


Fig. 43.14 A position–position architecture effectively creates a spring and damper between the two robots

may be entirely hidden from the user. Naturally this defeats the purpose of force feedback. In these cases, a local force control system may be used to render the slave back-drivable. Alternatively, a position–force architecture may be selected.

Position–Force Architecture

In the above position–position architecture, the user was effectively presented with the slave's controller force. While this is very stable, it also means the user feels the friction and inertia in the slave robot, which the controller is actively driving to overcome. In many scenarios this is undesirable. To avoid the issue, position–force architectures place a force sensor at the tip of the slave robot and feedback the force from there. That is, the system is controlled by

$$\begin{aligned} \boldsymbol{F}_\mathrm{m} &= \boldsymbol{F}_\mathrm{sensor}\,, \\ \boldsymbol{F}_\mathrm{s} &= -\boldsymbol{K}_\mathrm{s}(\boldsymbol{x}_\mathrm{s}-\boldsymbol{x}_\mathrm{sd})-\boldsymbol{B}_\mathrm{s}(\dot{\boldsymbol{x}}_\mathrm{s}-\dot{\boldsymbol{x}}_\mathrm{sd})\,. \end{aligned} \tag{43.8}$$

This allows the user to only feel the external forces acting between the slave and the environment and presents a more clear sense of the environment. However, this architecture is less stable: the control loop passes from master motion to slave motion to environment forces back to master forces. There may be some lag in the slave's motion tracking not to mention any delay in communications. Meanwhile the loop gain can be very high: a small motion command can turn into a large force if the slave is pressing against a stiff environment. In combination, stability may be compromised in stiff contact and many systems exhibit contact instability in these cases.

43.4.2 Passivity and Stability

The two basic architectures presented in Sect. 43.4.1 clearly illustrate one of the basic tradeoffs and challenges in force feedback: stability versus performance. Stability issues arise because any models of the system depend on the environment as well as the user. Both these elements are difficult to capture and, if we assume we want to explore unknown environments, impossible to predict. This issue makes a stability analysis very difficult. A common tool that avoids some of these issues is the concept of passivity. Although passivity provides only a sufficient (not a necessary) condition for stability, it incorporates the environmental uncertainly very well.

Passivity is an intuitive tool that examines the energy flows in a system and makes stability assertions if energy is dissipated instead of generated. Three rules are of importance here. First, a system is passive if and only if it cannot produce energy. That is the output energy from the system is limited by the initial and accumulated energy in the system. Second, two passive systems can be combined to form a new passive system. Third, the feedback connection of two passive systems is stable.

In the case of telerobotics, we generally assume that the slave robot will only interact with passive environments, that is, that the environments do not contain active motors or the like. Without the human operator, stability can therefore be assured if the system is also passive, without needing an explicit environment model.

On the master side the operator closes a loop and has to be considered in the stability analysis. In general, the master robot will be held by the user's hand and arm. A variety of models and parameters describe the human arm dynamics, mainly in the form of a mass–damper–spring system. In [43.60] we find a summary of model parameters used by different authors. For an impedance-controlled haptic interface, found in many systems, the operator adds some physical damping and a light touch ($\boldsymbol{F}_\mathrm{human} \approx 0$) is often one of the more challenging scenarios [43.61, 62]. As such some analyses ignore the human operator entirely. But even in general, humans seem to achieve stable interactions with all passive systems and passivity is usually considered sufficient for stable human–telerobot interactions.

To apply passivity, we take the system originally depicted in Fig. 43.13 and describe it as two-port elements in Fig. 43.15. Each port describes the energy flow between subsystems, where power is the product of a pair

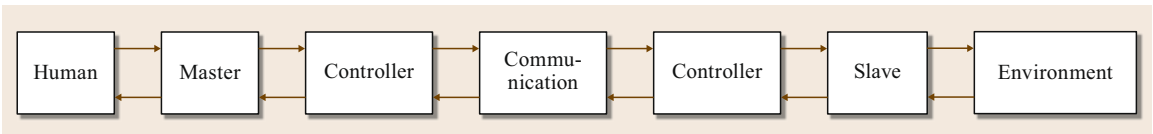


Fig. 43.15 A teleoperator can be analyzed as a chain of-two port elements connecting the one-port operator to the one-port environment

of collocated physical variables. We can choose a sign convention, for example declaring positive power to flow toward the remote environment. Then at the first boundary, the positive power flow is the product of master velocity $\dot{\boldsymbol{x}}_\mathrm{m}$ times applied (human) force $\boldsymbol{F}_\mathrm{human}$

$$P_\mathrm{left} = \dot{\boldsymbol{x}}_\mathrm{m}^\mathrm{T} \boldsymbol{F}_\mathrm{human} \,. \tag{43.9}$$

Meanwhile at the last boundary, the positive power flow is the product of the slave velocity $\dot{\boldsymbol{x}}_\mathrm{s}$ times the environment force $\boldsymbol{F}_\mathrm{env}$ (which ultimately opposes the human force)

$$P_\mathrm{right} = \dot{\boldsymbol{x}}_\mathrm{s}^\mathrm{T} \boldsymbol{F}_\mathrm{env} \,. \tag{43.10}$$

Therefore the entire telerobotic system is passive if

$$\begin{aligned} \int_0^t P_\mathrm{input}\,\mathrm{d}t &= \int_0^t (P_\mathrm{left} - P_\mathrm{right})\,\mathrm{d}t \\ &= \int_0^t \left(\dot{\boldsymbol{x}}_\mathrm{m}^\mathrm{T}\boldsymbol{F}_\mathrm{human} - \dot{\boldsymbol{x}}_\mathrm{s}^\mathrm{T}\boldsymbol{F}_\mathrm{env}\right)\,\mathrm{d}t \\ &> -E_\mathrm{store}(0) \,. \end{aligned} \tag{43.11}$$

Obviously, the ideal teleoperator ($\dot{\boldsymbol{x}}_\mathrm{m} = \dot{\boldsymbol{x}}_\mathrm{s}, \boldsymbol{F}_\mathrm{human} = \boldsymbol{F}_\mathrm{env}$) is passive. These definitions can also be generalized to six dimensional twists and wrenches.

To simplify the analysis, we can examine the passivity of each two-port element or subsystem and then deduce the overall passivity. The master and slave robots are mechanical elements and hence passive. The controllers of a position–position architecture mimic a spring and damper, which are also passive elements. So without delay and ignoring discretization, quantization, and other unmodelled effects, a simple position–position architecture is passive. The following sections will address some of these limitations and thus create controllers that are passive under more circumstances.

While powerful to handle uncertainty, passivity can be overly conservative. Many controllers are overdamped if every subsystem is passive. In contrast, the combination of an active and a passive subsystem may be passive and stable and show less dissipation. This is particularly true for the cascaded arrangement of two-port elements in the telerobotic system of Fig. 43.15. From network theory, the Llewellyn criterion specifies when a possibly active two-port connected with any passive one-port becomes passive. This two-port is then labeled unconditionally stable, as it will be stable in connection to any two passive one-ports. The Llewellyn criterion may hence be used as a more general stability test for telerobotic systems or components [43.63].

Passive controllers are also limited as they cannot hide the dynamics of the slave robot. In the above position–position architecture, the user will feel the forces associated with the slave inertia. In contrast the position–force architecture hides the slave inertia and friction from the user. As such, when the user inserts kinetic energy into the master without feeling any resistance, the system itself creates and injects the kinetic energy for the slave. This violates passivity and provides another insight as to why the architecture suffers from potential stability problems.

43.4.3 Transparency and Multichannel Feedback

Both basic architectures can be captured by the general teleoperator control system described by *Lawrence* [43.13], and later expanded by *Hashtrudi-Zaad* and *Salcudean* [43.63] and shown in Fig. 43.16. Ideally a system will equalize both the operator and environment forces as well as the master and slave motions. Therefore, it is desirable to measure both force and velocity (from which position may be integrated or vice versa) at both sites. With this complete information, the slave may, for example, start moving as soon

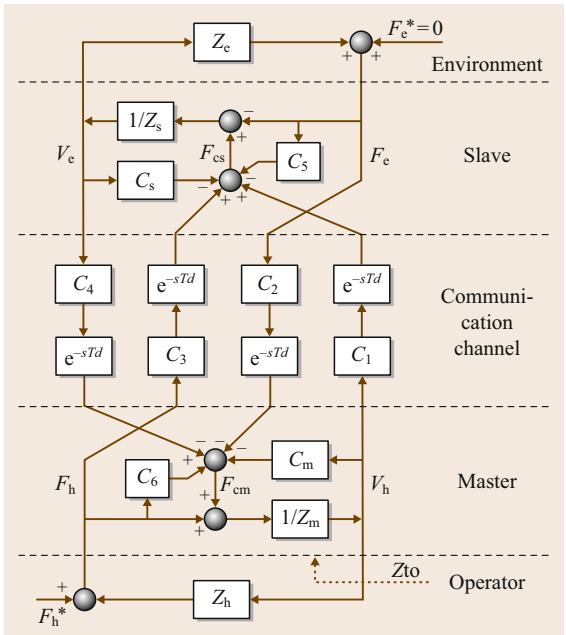


Fig. 43.16 In general, a controller will use both position and force information from both master and slave robot (after [43.63], adapted from [43.13])

Avidbots Ex1016
Page 1126 of 2154

as the user applies a force to the master, even before the master itself has moved.

Following these concepts derived in [43.13], we can examine the relationships between velocity and force, in the form of impedances and admittances. Note we do this in a single degree of freedom, assuming that all degrees of freedom may be treated independently. The environment will exhibit some impedance $Z_e(s)$ that is not known in advance and relates the environment force to the slave's velocity

$$\boldsymbol{F}_e(s) = Z_e(s)\boldsymbol{v}_s(s)\,. \tag{43.12}$$

If we describe the teleoperator in whole as a two-port with a hybrid matrix formulation

$$\begin{pmatrix} F_h(s) \\ \boldsymbol{v}_m(s) \end{pmatrix} = \begin{pmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_s(s) \\ -F_e(s) \end{pmatrix}, \tag{43.13}$$

then the user will perceive the impedance

$$Z_{to}(s) = \frac{\boldsymbol{F}_h(s)}{\boldsymbol{v}_m(s)} = (H_{11} - H_{12}Z_e)(H_{21} - H_{22}Z_e)^{-1}\,. \tag{43.14}$$

Transparency describes how close the user's perceived impedance comes to recreating the true environment impedance.

For a detailed treatment of passivity in telerobotics, impedance and admittance interpretations and designs, and transparency, we refer to some of the seminal works in [43.11–13, 63–66].

43.4.4 Time Delay and Scattering Theory

When delays occur in the communications between the local and remote site, even position–position architectures can suffer from serious instabilities [43.67, 68]. This can be traced to the communications block in Fig. 43.15, where the power entering the left side and exiting the right side do not add up. Rather energy may be generated inside the block, which feeds the instability [43.9].

Several approaches to operate under delay have been studied [43.69], in particular shared compliant control [43.70] and the addition of local force loops [43.71]. The use of the Internet for communication, adding variability to the delay, is also an area of interest [43.72, 73]. This further evokes issues of data reduction [43.74].

Here we note that natural wave phenomena are bilateral passive elements that tolerate delay. If the control system is described in the frequency domain and scattering matrices are used in place of impedance and admittance matrices, the system can tolerate delays [43.75]. Scattering matrices relate the sum of velocity and force to their difference, so that passivity becomes a condition on the system gain, which is unaffected by the delay.

43.4.5 Wave Variables

Building on the realization that delay communications can be active and that wave phenomena circumvent the issue, wave variables provide an encoding scheme that is tolerant of delay [43.76]. Consider the power flowing through the system and separate the power moving forward and returning.

$$P = \dot{\boldsymbol{x}}^{\mathrm{T}}\boldsymbol{F} = \frac{1}{2}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} - \frac{1}{2}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v} = P_{\text{forward}} - P_{\text{return}}\,, \tag{43.15}$$

where the forward and returning power are by construction nonnegative. This leads to the definition of the wave variables

$$\boldsymbol{u} = \frac{b\dot{\boldsymbol{x}} + \boldsymbol{F}}{\sqrt{2b}}\,, \quad \boldsymbol{v} = \frac{b\dot{\boldsymbol{x}} - \boldsymbol{F}}{\sqrt{2b}}\,, \tag{43.16}$$

where $\boldsymbol{u}$ is the forward-moving and $\boldsymbol{v}$ the returning wave.

If velocity and force signals are encoded into wave variables, transmitted across the delay, and decoded at the far site, the system remains passive regardless of delay. In fact, in the wave domain, passivity corresponds to a wave gain of less than or equal to unity. No requirements are placed on phase and so lag does not destroy stability.

The wave impedance b relates velocity to force and provides a tuning knob to the operator. Large b values mean the system increases force feedback levels at the cost of feeling high inertial forces. Small values of b lower any unwanted sensations, making it easy to move quickly, but also lower the desirable environment forces. Ideally the operator would lower b when there is no risk of contact and raise b when contact is imminent. The concept of scattering has also been extended to a coordinate free context [43.77].

Recent developments are incorporating both position–position and position–force architectures within the wave frame work, so the resulting systems are stable with any environment, stable with any delay, yet maintain the feedback of high-frequency forces that help the operator identify happenings at the remote site [43.78]. To improve performance and assist the operator, especially across the Internet, predictors may also be incorporated [43.79].

Avidbots Ex1016
Page 1127 of 2154

43.4.6 Teleoperation with Lossy Communication

The Internet provides an affordable and ubiquitously accessible communication medium. However, it can also introduce nondeterministic effects and lossy connections due to time-varying delays, packet losses, and packet reordering. How to achieve passive bilateral teleoperation over such lossy communication network has been an active research topic in telerobotics.

Many telerobotic systems have relied on the position-position architecture (43.7) with extra damping injection. This leads to the following proportional-derivative (PD) control law, with a simple structure and explicit position feedback,

$$\begin{aligned} \boldsymbol{F}_{\mathrm{m}} &= -\boldsymbol{B}_{\mathrm{d}}\dot{\boldsymbol{x}}_{\mathrm{m}}(t) - \boldsymbol{B}[\dot{\boldsymbol{x}}_{\mathrm{m}}(t) - \dot{\boldsymbol{x}}_{\mathrm{s}}(t-\tau_1)] \\ &\quad - \boldsymbol{K}[\boldsymbol{x}_{\mathrm{m}}(t) - \boldsymbol{x}_{\mathrm{s}}(t-\tau_1)] \\ \boldsymbol{F}_{\mathrm{s}} &= -\boldsymbol{B}_{\mathrm{d}}\dot{\boldsymbol{x}}_{\mathrm{s}}(t) - \boldsymbol{B}[\dot{\boldsymbol{x}}_{\mathrm{s}}(t) - \dot{\boldsymbol{x}}_{\mathrm{m}}(t-\tau_2)] \\ &\quad - \boldsymbol{K}[\boldsymbol{x}_{\mathrm{s}}(t) - \boldsymbol{x}_{\mathrm{m}}(t-\tau_2)]\,, \end{aligned} \tag{43.17}$$

where $\boldsymbol{B}_{\mathrm{d}}, \boldsymbol{K}, \boldsymbol{B}$ are the stabilizing absolute damping and the PD control gains, and $\tau_1, \tau_2 \geq 0$ are the communication delays from slave to master and from master to slave, respectively.

The usage of this controller across lossy communication was yet hampered (or at least reserved) due to the lack of theoretical guarantees for its passivity and stability. Anecdotes say that with sufficiently large damping and small delays and PD gains, the closed-loop system remains stable. This anecdotal observation was justified in [43.80] for the case of constant delay, that is, the PD-like controller (43.17) is passive if the following condition is met,

$$\boldsymbol{B}_{\mathrm{d}} > \frac{\bar{\tau}_1 + \bar{\tau}_2}{2}\boldsymbol{K}\,, \tag{43.18}$$

where $(\bar{\tau}_1 + \bar{\tau}_2)/2$ is the upper-bound of the round-trip delay, which can typically be estimated easily. Without human or environment forcing, master and slave positions will also converge to each other. This result was extended in [43.81] for the case of time-varying delay. Passivity of the PD-like controller (43.17) is guaranteed if

$$\boldsymbol{B}_{\mathrm{d}} > \frac{\sqrt{\bar{\tau}_1^2 + \bar{\tau}_2^2}}{2}\boldsymbol{K} \quad \text{and} \quad |\dot{\tau}_i(t)| < 1\,, \tag{43.19}$$

where the second condition means the delay $\tau_i(t)$ neither grows nor decreases faster than time t. We refer to [43.81] for cases with asymmetric controller gains and to [43.82, 83] for extensions to general digital lossy communication networks.

With a fixed structure, the PD controller (43.17) has to be tuned to the worst-case conditions according to (43.18) and (43.19). It may thus exhibit drastic performance degradation for severely variable communications. To overcome such fixed structure limitation, several passivity-enforcing flexible control techniques have been recently proposed.

The technique of passivity-observer/passivity-controller (PO/PC) was originally devised for haptic device control [43.84] and has been extended for the bilateral teleoperation with digital network with time-varying delay [43.85–87]. Each PO does real-time bookkeeping of energy flows at the master and slave sites. The PC is activated to dissipate energy whenever a passivity violation is detected. The passive set-position modulation (PSPM) framework was proposed in [43.88], where the desired set-position signal received from a general digital lossy communication network is real-time modulated as close to the original set-position signal as possible, yet, only to the extent permissible by the available energy in the system. Passification of a desired control force utilizing the device's physical damping was addressed in [43.89] under the name of energy bounding algorithm (EBA). The idea of modulating a control signal or action under the passivity constraint was also adopted in the two-layer approach [43.90], where the transparency-layer is designed for best performance, while the passivity-layer superimposes constraints to enforce passivity.

Overall, the PO/PC approach may be considered as *corrective* (i. e., detect violation of passivity first and then apply passifying action) whereas the PSPM, EBA and the two-layer approaches are *preventive* (i. e., modulate/bound control action to prevent passivity violation). Interestingly, all the PO/PC, PSPM, EBA and two-layer approaches share some common characteristics: 1) they transmit energy packets along with other information over the communication network to replenish energy levels and enable useful work; and 2) each of these approaches activates their passifying action only when necessary, thus, can significantly improve control performance compared to fixed-structure teleoperation controllers, while also robustly enforcing passivity against a variety of communication imperfectness.

43.4.7 Port-Based Approaches

From Sect. 43.4.2 we know that energy flows provide a great description of telerobotic systems that physically interact with unknown environments and a human user. Indeed all physical interaction dynamics are fundamentally bound to energy exchanges. Passivity is a well suited analysis tool and assures stability if the system energy is bounded. All possible instabilities

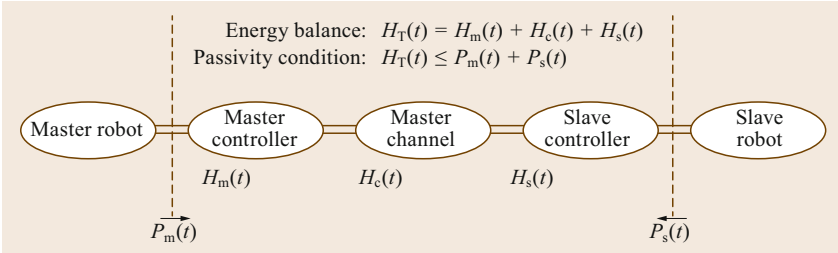


Fig. 43.17 Energy balance of the telemanipulation chain (after [43.90])

can be traced to unsupervised energy injections via the actuators. In Sect. 43.4.6 we also saw methods that explicitly monitor energy flows. Here we describe port-based approaches designed explicitly around energy exchanges. This can be advantageous to handle nonlinear system dynamics, discretization, time-varying delays, and other issues.

The concept of power ports and energy flows (Sect. 43.4.2) allows a precise analytical formulation of physical systems. This approach, which stems from concepts used in bond-graphs, enables a pure energy based analysis of complex nonlinear physical systems [43.91]. It has also provided new perspectives on modeling and robot control [43.92].

To handle sampling issues and prevent instabilities due to limited sampling rates, consider an energy-consistent discretization [43.93]. Rather than measuring power flow at discrete intervals, consider the energy flow that occurs continuously over the entire sampling interval. An exact measure of the energy transfer between sample time kT and the following sample time $(k+1)T$ is

$$\Delta E_k = \int_{kT}^{(k+1)T} \tau(t)\dot{q}(t)\,\mathrm{d}t\,. \tag{43.20}$$

Assume an electric motor with an ideal current amplifier and without any commutation effects is generating the torque as commanded by a zero order hold (ZOH) digital to analog converter. Further assume the position sensor is collocated and synchronized to the ZOH transitions. The torque may then be considered constant during the interval, so that

$$\Delta E_k = \int_{kT}^{(k+1)T} \tau_{k,k+1}\dot{q}(t)\,\mathrm{d}t\,. \tag{43.21}$$

Taking the torque out of the integral, we realize that

$$\Delta E_k = \tau_{k,k+1}[q(k+1) - q(k)]\,. \tag{43.22}$$

This simple result has far reaching consequences for interfacing the digital and physical world and can be easily calculated even if the sampling time varies VIDEO 724. And some of the assumptions may be relaxed with suitable adaptation. Using this more precise measurement of energy flow leads to more consistent energy book-keeping.

As it is shown in Fig. 43.17, we can now track energy in the digital world, associating individual available energy reservoirs H_m and H_s with the master and slave controllers. The communication channel transmits both data and energy packets (EPs), where EPs contain only information about an energy quanta. An EP is only sent if the transmitter has sufficient available energy, which is then decreased by the quanta. An arriving EP injects the quanta into the receiver's available energy. In this way the total *virtual* energy in the system will never increase. If the communication protocol allows an EP to be lost, this will remove energy similar to dissipation. This process is independent of any constant or variable delay.

The port-based paradigm allows any nonlinear control algorithm. But any applied control force will have an energetic consequence and will be allowed if and only if the associated available energy level is sufficiently high. Following the classification of Sect. 43.4.6, this paradigm is preventive. It prevents energy generation without needing to dissipate unexpected energy appearances.

Strategies are needed to address the exchange of energy between master and slave. A simple protocol introduced in [43.90] continually transmits EPs with a percentage of the locally stored energy. It can be shown that this will result in an equal distribution of energy between the master and slave controller. Also, if necessary, a small damper can be superimposed at the master side to extract energy from the human as needed.

With the port-based paradigm, the data communication (related to transparency) and the energy communication (related to passivity) are split: controllers may be nonlinear, energy and data transmission may be independent, and there is basically no restriction on the kind of controller which can be implemented. The fact that energy and data are separated gives the name to the two-layer approach [43.90].

Avidbots Ex1016
Page 1129 of 2154

43.5 Emerging Applications of Telerobotics

Historically, telerobotics research has focused on a conventional setup with two fixed-based robotic manipulators serving as the master and slave devices. Recently, there have been substantial efforts to extend the telerobotic theories and frameworks to more unconventional scenarios. Here we summarize some recent results on these emerging applications. The summary is by no means exhaustive and, to be consistent with the chapter, focuses on controls aspects and providing a stable bilateral user interface.

43.5.1 Telerobotics for Mobile Robots

Mobile robots are useful slave devices if the task covers a large spatial area. Flying robots, in particular, can operate in three-dimensional space without being bound to the ground. For mobile robot teleoperation, force feedback may be used to convey proprioceptive information of the slave robot (e.g., velocity), or haptic feedback of virtual (or real) objects in the remote environment.

A key difference of mobile and flying robot teleoperation compared to a conventional setup is *kinematic dissimilarity* [43.94]: the workspace of the master device is bounded while the workspace of the slave robot is unbounded. This suggests to couple the master position to the slave velocity, as with rate-control described in Sect. 43.3.3. A direct coupling between master position and slave velocity, however, cannot be addressed by the standard passivity framework (Sect. 43.4.2). The master position and the slave velocity possess different relative degrees with respect to the torque. One way to circumvent this difficulty is to utilize so called r-variable

$$r := \dot{q} + \lambda q \,. \tag{43.23}$$

That is, by utilizing inverse-dynamics similar to adaptive control design [43.95, 96] or by injecting some PD-type local state feedback with redefined output r [43.97] (Fig. 43.18), it is possible to render the master device to be passive with this r-variable replacing the velocity $\dot{q}$ in its original passivity relation (43.9). This implies that we can couple the r-variable and the slave's velocity passively, just like standard passivity-based controllers.

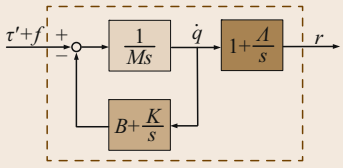


Fig. 43.18 Feedback r-passivation by using PD-type state feedback

Meanwhile the r-variable contains the master position information.

We can also achieve passive mobile robot teleoperation with time-varying delays by extending the port-based ideas presented in Sect. 43.4.7 with the concept of a *virtual vehicle* [43.98]. This virtual vehicle is a simulated slave system, evolving in a gravitation-free field and having a finite energy tank. Commands from the master are only allowed to transfer energy from the tank to accelerate the vehicle or to return energy by decelerating the vehicle. This creates a closed energy system for the slave. A passive behavior can be achieved with a viscoelastic connection between the virtual and real vehicles.

Mobile telerobotics often uses nonholonomically-constrained wheeled mobile robots (WMRs) with pure-rolling wheels. For slave WMRs, it is often possible (with some low-level control as stated below) to split motions (i. e., velocity directions) into those requiring a position-velocity coupling as stated above (e.g., forward velocity of WMR), and others that may be controlled by a standard position-position coupling (e.g., rotation angle of WMR) as explained in Sects. 43.4.1 and 43.4.6.

Other mobile robots, particularly quadrotor or ducted-fan type aerial robots (Chap. 36) or thrust-propelled autonomous aquatic vehicles (AUVs, Chap. 25), are under-actuated with fewer control variables than degrees of freedom and also defy standard teleoperation techniques. To address under-actuation, *abstraction* of the slave robot has been utilized [43.98, 99], that is, human users telecontrol a fully-actuated *virtual system*, assuming that its motion is adequately describing that of the real slave robot, while a certain low-level tracking control is employed to drive the under-actuated slave mobile robot to tightly follow this virtual system. This abstraction leads to a hierarchical control design, composed of: 1) a high-level teleoperation control layer between the virtual vehicle and the master device; and 2) a low-level tracking control layer, into which the issue of the slave's under-actuation is confined.

For the teleoperation layer, we may then use the conventional teleoperation techniques explained in Sect. 43.4 with the (r, v)-coupling as explained above. We may also use the recently proposed control techniques explained in Sect. 43.4.6, which promise sharper haptic feedback with guaranteed stability against lossy communication. Presenting both the slave robot's proprioceptive information and the presence of surrounding objects for obstacle avoidance via the same haptic feedback channel, however, typically results in

Avidbots Ex1016
Page 1130 of 2154

a perceptual ambiguity [43.100]. A further perceptual modality (e.g., vision) is usually necessary to resolve this ambiguity. In some applications, the slave mobile robots are required to move around in but to not directly contact any physical environments. In these cases, virtual forces can be generated for any obstacles and the slave robot is interacting only with this precisely known virtual force field. This suggests it would be sufficient to enforce slave stability versus slave passivity ([43.99], VIDEO 71 and VIDEO 72), as a less conservative controller will likely give better system performance.

43.5.2 Multilateral Telerobotics

Many practical telerobotic tasks require dexterous, complicated, and large degree-of-freedom motions, e.g., in surgical training, rehabilitation, or exploration. For such tasks, we may utilize a team of multiple cooperative slave robots or a single slave robot possessing many degrees of freedom. The complexity involved in both cases may require multiple human operators to adequately control and coordinate all degrees of freedom. Following [43.102], we thus consider the following scenarios:

1. Single-master multiple-slave (SMMS) systems
2. Multiple-master multiple-slave (MMMS) systems
3. Multiple-master single-slave (MMSS) systems
4. Single-master single-slave (SMSS) systems, which constitute the conventional telerobotic setup.

Here we introduce some recent results applicable to SMMS and MMSS systems.

It is common for single-master multiple-slave (SMMS) systems to autonomously control simple subtasks among the slaves, e.g., maintaining a grasp, maintaining connectivity, or avoiding collisions (Fig. 43.19). In particular, in [43.101, 104, 105], passive decomposition [43.106] is utilized to decompose the dynamics of multiple slave robots into their shape system, describing the inter-slave formation aspects, and the locked system, abstracting their collective motion and centroid behavior. The locked system can be telecontrolled by a single human user, while an autonomous controller regulates the shape system to maintain the cooperative grasp of an object between the slaves.

Another interesting development in SMMS telerobotics is the combination with the frameworks of multiagent cooperative control (i. e., consensus, flocking, synchronization, and other behaviors). For instance, in [43.103] and VIDEO 73, a single human user directly telecontrols a single leader agent among the slaves, while the behavior of the other slaves is dictated by a leader-follower information graph (Fig. 43.20). A time-varying graph topology with arbitrary split/join operations among the slaves allows for reconfiguration, e.g., for navigation in cluttered environments. And the concept of virtual energy tanks, along with the port-Hamiltonian modeling and port-based approach (Sect. 43.4.7), is utilized to enforce passivity and stability of the total system.

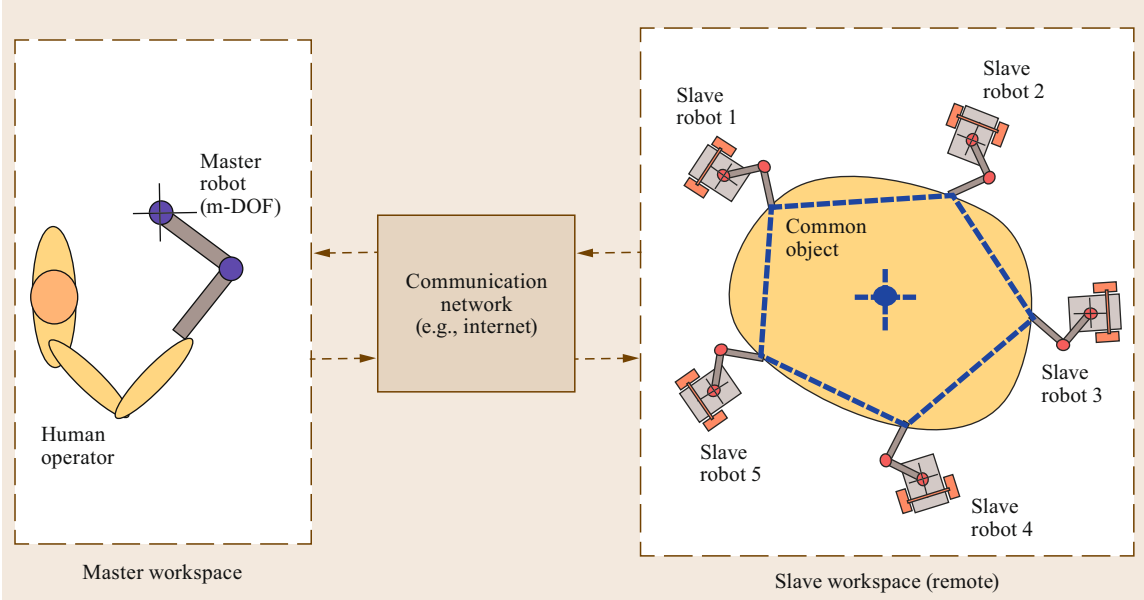


Fig. 43.19 SMMS telerobotic control of multiple slave robots (after [43.101])

Avidbots Ex1016
Page 1131 of 2154

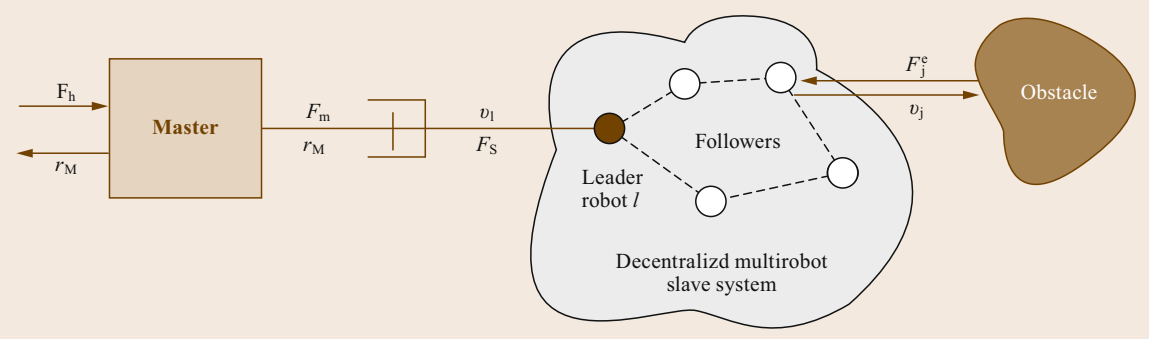


Fig. 43.20 SMMS telerobotic control architecture with possibly time-varying leader-follower information topology (after [43.103])

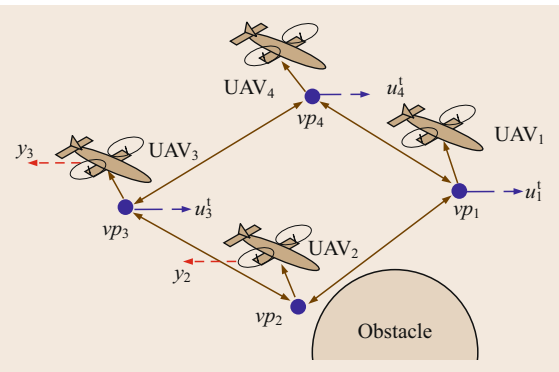


Fig. 43.21 SMMS control architecture for multiple unmanned aerial vehicles (UAVs; after [43.99]), where a single user telecontrols some of the virtual systems (via u_1^t, u_4^t) while telesensing the state of some of real UAVs (i. e., y_2, y_3), which are low-level controlled to follow their respective virtual systems

A distributed SMMS approach was also proposed in [43.99], in VIDEO 71 and VIDEO 72, to enable a single human user to telecontrol some of the slave robots. The inter-slave behaviors are encoded via a distributed artificial potential constructed on a time-invariant undirected information graph (Fig. 43.21). By using kinematic virtual systems to abstract the slave robots, the work achieves the combination of master-passivity and slave-stability. It guarantees no collisions among the slave robots or with obstacles and no separations among the slave robots.

A control approach for MMSS telerobotic systems was proposed in [43.107] and in VIDEO 75, where two human users telecontrol distinct frames on a single large-DOF slave robot. The velocity space of the slave is decomposed according to the two command motions, resolving conflicts between and constraints imposed on them. Priority is given to the primary user's commands

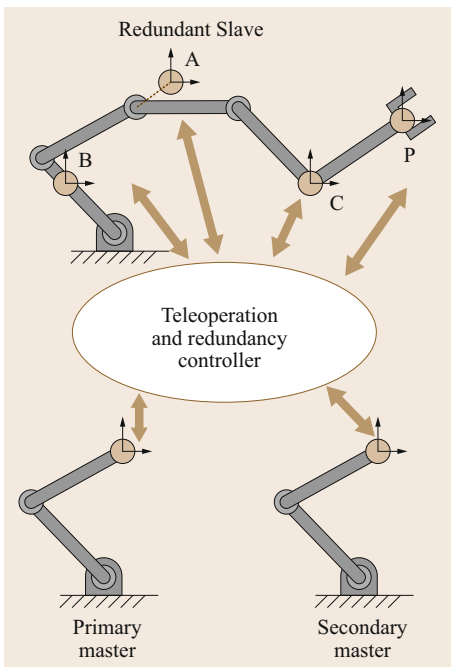


Fig. 43.22 Trilateral teleoperation (after [43.107]), where the primary master controls the end-effector frame P, while the secondary master controls task-space frame A, B or C: if A is chosen, primary and secondary tasks are nonconflicting/nonconstrained; if B is chosen, nonconflicting, yet, the secondary task is constrained; if C is chosen, conflicting, thus, prioritized nullspace control is necessary between primary and secondary tasks

(Fig. 43.22). The kinematics-level prioritized velocity commands are realized by dynamics-level adaptive control for all master and slave robots.

A different shared trilateral MMSS teleoperation framework was developedin [43.108], where two human users teleoperate the *same* point of a single slave

Avidbots Ex1016
Page 1132 of 2154

robot. Their control authority is adjusted with a dominance factor $\alpha \in [0, 1]$ set according to the task objective (e.g., $\alpha = 1$ for training or $\alpha = 0$ for evaluation). The MMSS system is optimized for a measure of transparency and Llewellyn's criteria is applied to the equivalent two-port system (with environment impedance Z_e embedded) to establish unconditional stability.

These multilateral and the previous mobile telerobotic approaches are promising to significantly expand the utility and application horizons of conventional telerobotics. Significant technical challenges and research questions remain, for example:

1. How to assign and determine the roles for multiple coordinating human users?
2. How to systematically split control tasks into teleoperative and autonomous control?
3. What are the most suitable performance measures, which are likely different from the conventional metrics?
4. What is the best form of human interface (or haptic feedback) [43.100] for the mobile and multilateral telerobotics and how to complement the interface with other modality (e.g., visual feedback).

43.6 Conclusions and Further Reading

Despite its age, telerobotics remains an exciting and vibrant area of robotics. In many ways, it forms a platform which can utilize the advances in robotic technologies while simultaneously leveraging the proven skills and capabilities of human users. Compare this, for example, with the development of the automobile and its relation to the driver. As cars are gradually becoming more sophisticated with added electronic stability control and navigation systems, they are becoming safer and more useful to their operators, not replacing them. Similarly telerobotics serves as a pathway for gradual progress and, as such, is perhaps best suited to fulfill robotics long-held promise of improving human life. It is seeing use in the challenging area of search and rescue. And with the recent developments and commercializations in telerobotic surgery systems, it is indeed impacting on the lives of tens of thousands of patients in a profound fashion and extending the reach of robotics into our world.

For further reading in the area of supervisory control, we refer to *Sheridan* [43.39]. Though published in 1992, it remains the most complete discussion on the topic. Unfortunately few other books are devoted to or even fully discuss telerobotics. In [43.109] many recent advances, including methods, experiments, applications, and developments, are collected. Beyond this, in the areas of bilateral and shared control, as well as to understand the various applications, we can only refer to the citations provided. Finally, in addition to the standard robotics journals, we note in particular *Presence: Teleoperators and Virtual Environments*, published by the MIT Press. Combined with virtual-reality applications, it focuses on technologies with a human operator.

Video-References

VIDEO 71 Semi-autonomous teleoperation of multiple UAVs: Passing a narrow gap
available from http://handbookofrobotics.org/view-chapter/43/videodetails/71

VIDEO 72 Semi-autonomous teleoperation of multiple UAVs: Tumbing over obstacle
available from http://handbookofrobotics.org/view-chapter/43/videodetails/72

VIDEO 73 Bilateral teleoperation of multiple quadrotors with time-varying topology
available from http://handbookofrobotics.org/view-chapter/43/videodetails/73

VIDEO 74 Passive teleoperation of nonlinear telerobot with tool-dynamics rendering
available from http://handbookofrobotics.org/view-chapter/43/videodetails/74

VIDEO 75 Asymmetric teleoperation of dual-arm mobile manipulator
available from http://handbookofrobotics.org/view-chapter/43/videodetails/75

VIDEO 297 Tele-existence master–slave system for remote manipulation
available from http://handbookofrobotics.org/view-chapter/43/videodetails/297

VIDEO 298 JPL dual-arm telerobot system
available from http://handbookofrobotics.org/view-chapter/43/videodetails/298

VIDEO 299 Single and dual arm supervisory and shared control
available from http://handbookofrobotics.org/view-chapter/43/videodetails/299

VIDEO 318 Teleoperated humanoid robot – HRP
available from http://handbookofrobotics.org/view-chapter/43/videodetails/318

VIDEO 319 Teleoperated humanoid robot – HRP: Tele-driving of lifting vehicle
available from http://handbookofrobotics.org/view-chapter/43/videodetails/319

Avidbots Ex1016
Page 1133 of 2154

VIDEO 321 Multi-modal multi-user telepresence and teleaction system
available from http://handbookofrobotics.org/view-chapter/43/videodetails/321

VIDEO 322 Laparoscopic telesurgery workstation
available from http://handbookofrobotics.org/view-chapter/43/videodetails/322

VIDEO 724 Passivity of IPC strategy at 30 Hz sample rate
available from http://handbookofrobotics.org/view-chapter/43/videodetails/724

References

43.1 M. Buss, G. Schmidt: Control problems in multi-modal telepresence systems. In: *Advances in Control*, ed. by P.M. Frank (Springer, London 1999) pp. 65–101

43.2 W.R. Ferell, T.B. Sheridan: Supervisory control of remote manipulation, IEEE Spectrum **4**(10), 81–88 (1967)

43.3 R.C. Goertz: Fundamentals of general-purpose remote manipulators, Nucleonics **10**(11), 36–42 (1952)

43.4 R.C. Goertz: Mechanical master–slave manipulator, Nucleonics **12**(11), 45–46 (1954)

43.5 R.C. Goertz, F. Bevilacqua: A force-reflecting positional servomechanism, Nucleonics **10**(11), 43–45 (1952)

43.6 W.R. Ferell: Remote manipulation with transmission delay, IEEE Trans. Hum. Factors Electron. **6**, 24–32 (1965)

43.7 T.B. Sheridan, W.R. Ferell: Remote manipulative control with transmission delay, IEEE Trans. Hum. Factors Electron. **4**, 25–29 (1963)

43.8 F. Miyazaki, S. Matsubayashi, T. Yoshimi, S. Arimoto: A new control methodology towards advanced teleoperation of master–slave robot systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 997–1002

43.9 R.J. Anderson, M.W. Spong: Asymptotic stability for force reflecting teleoperators with time delay, Int. J. Robotics Res. **11**(2), 135–149 (1992)

43.10 G. Niemeyer, J.-J.E. Slotine: Stable adaptive teleoperation, IEEE J. Ocean. Eng. **16**(1), 152–162 (1991)

43.11 J.E. Colgate: Robust impedance shaping telemanipulation, IEEE Trans. Robotics Autom. **9**(4), 374–384 (1993)

43.12 B. Hannaford: A design framework for teleoperators with kinesthetic feedback, IEEE Trans. Robotics Autom. **5**(4), 426–434 (1989)

43.13 D.A. Lawrence: Stability and transparency in bilateral teleoperation, IEEE Trans. Robotics Autom. **9**(5), 624–637 (1993)

43.14 D. Kuban, H.L. Martin: An advanced remotely maintainable servomanipulator concept, Proc. 1984 Natl. Top. Meet. Robotics Remote Handl. Hostile Environ., Washington (1984)

43.15 J. Vertut, P. Coiffet: *Teleoperation and Robotics: Evolution and Development* (Kogan Page, London 1985)

43.16 J. Vertut: MA23M contained servo manipulator with television camera, PICA and PIADE telescopic supports, with computer-integrated control, Proc. 28th Remote Syst. Technol. Conf. (1980) pp. 13–19

43.17 J. Vertut, P. Coiffet: Bilateral servo manipulator MA23 in direct mode and via optimized computer control, Proc. 2nd Retote Manned Syst. Technol. Conf. (1977)

43.18 A.K. Bejczy: Towards advanced teleoperation in space, Prog. Astronaut. Aeronaut **161**, 107–138 (1994)

43.19 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: Sensor-based space robotics – ROTEX and its telerobotic features, IEEE Trans. Robotics Autom. **9**(5), 649–663 (1993)

43.20 T.H. Massie, J.K. Salisbury: The phantom haptic interface: A device for probing virtual objects, Proc. ASME Int. Mech. Eng. Congr. Exhib., Chicago (1994) pp. 295–302

43.21 P.S. Green, J.W. Hill, J.F. Jensen, A. Shah: Telepresence Surgery, IEEE Eng. Med. Bio. Mag. **14**(3), 324–329 (1995)

43.22 R.H. Taylor, J. Funda, B. Eldridge, S. Gomory, K. Gruben, D. LaRose, M. Talamini, L. Kavoussi, J. Anderson: A telerobotic assistant for laparoscopic surgery, IEEE Eng. Med. Bio. Mag. **14**(3), 279–288 (1995)

43.23 A.J. Madhani, G. Niemeyer, J.K. Salisbury: The black falcon: A teleoperated surgical instrument for minimally invasive surgery, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Victoria (1998) pp. 936–944

43.24 S. Charles, H. Das, T. Ohm, C. Boswell, G. Rodriguez, R. Steele, D. Istrate: Dexterity-enhanced telerobotic microsurgery, Proc. Int. Conf. Adv. Robotics (1997) pp. 5–10

43.25 G.S. Guthart, J.K. Salisbury: The Intuitive$^{\ddot{U}}$ telesurgery system: Overview and application, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 618–621

43.26 J.M. Sackier, Y. Wang: Robotically assisted laparoscopic surgery: From concept to development, Surg. Endosc. **8**(1), 63–66 (1994)

43.27 J. Marescaux, J. Leroy, F. Rubino, M. Vix, M. Simone, D. Mutter: Transcontinental robot assisted remote telesurgery: Feasibility and potential applications, Ann. Surg. **235**, 487–492 (2002)

43.28 G.H. Ballantyne: Robotic surgery, telerobotic surgery, telepresence, and telementoring – Review of early clinical results, Surg. Endosc. **16**(10), 1389–1402 (2002)

43.29 D.H. Birkett: Electromechanical instruments for endoscopic surgery, Minim. Invasive Ther. Allied Technol. **10**(6), 271–274 (2001)

43.30 J. Rosen, B. Hannaford: Doc at a distance, IEEE Spectrum **8**(10), 34–39 (2006)

Avidbots Ex1016
Page 1134 of 2154

43.31 A.M. Okamura: Methods for haptic feedback in teleoperated robot-assisted surgery, Ind. Robot **31**(6), 499–508 (2004)

43.32 T. Ortmaier, B. Deml, B. Kübler, G. Passig, D. Reintsema, U. Seibold: Robot assisted force feedback surgery, Springer Tracts Adv. Robotics **31**, 361–379 (2007)

43.33 B.M. Yamauchi: PackBot: A versatile platform for military robotics, Proc. SPIE **5422**, 228–237 (2004)

43.34 R.R. Murphy: Trial by fire [rescue robots], IEEE Robotics Autom. Mag. **11**(3), 50–61 (2004)

43.35 J. Wright, A. Trebi-Ollennu, F. Hartman, B. Cooper, S. Maxwell, J. Yen, J. Morrison: Driving a rover on mars using the rover sequencing and visualization program, Int. Conf. Instrumentation, Control Inf. Technol. (2005)

43.36 G. Hirzinger, K. Landzettel, D. Reintsema, C. Preusche, A. Albu-Schäffer, B. Rebele, M. Turk: ROKVISS – Robotics component verification on ISS, Proc. 8th Int. Symp. Artif. Intell. Robotics Autom. Space (iSAIRAS) (2005), Session2B

43.37 C. Preusche, D. Reintsema, K. Landzettel, G. Hirzinger: ROKVISS – Preliminary results for telepresence mode, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4595–4601

43.38 G. Hirzinger, K. Landzettel, B. Brunner, M. Fischer, C. Preusche, D. Reintsema, A. Albu-Schäffer, G. Schreiber, M. Steinmetz: DLR's robotics technologies for on-orbit servicing, Adv. Robotics **18**(2), 139–174 (2004)

43.39 T.B. Sheridan: *Telerobotics, Automation and Human Supervisory Control* (MIT Press, Cambridge 1992)

43.40 G. Hirzinger, J. Heindl, K. Landzettel, B. Brunner: Multisensory shared autonomy – A key issue in the space robot technology experiment ROTEX, Proc. RSJ/IEEE Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 221–230

43.41 B. Brunner, K. Arbter, G. Hirzinger: Task directed programming of sensor based robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1080–1087

43.42 A.K. Bejczy, W.S. Kim: Predictive displays and shared compliance control for time-delayed telemanipulation, Proc. IEEE/RSJ Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 407–412

43.43 P. Backes, K. Tso: UMI: An interactive supervisory and shared control system for telerobotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 1096–1101

43.44 L. Conway, R. Volz, M. Walker: Tele-autonomous systems: Methods and architectures for intermingling autonomous and telerobotic technology, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 1121–1130

43.45 S. Hayati, S.T. Venkataraman: Design and implementation of a robot control system with traded and shared control capability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1310–1315

43.46 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: ROTEX – The first remotely controlled robot in space, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 2604–2611

43.47 W.B. Griffin, W.R. Provancher, M.R. Cutkosky: Feedback strategies for telemanipulation with shared control of object handling forces, Presence **14**(6), 720–731 (2005)

43.48 T. Ortmaier, M. GrÖger, D.H. Boehm, V. Falk, G. Hirzinger: Motion estimation in beating heart surgery, IEEE Trans. Biomed. Eng. **52**(10), 1729–1740 (2005)

43.49 L. Rosenberg: Virtual fixtures: Perceptual tools for telerobotic manipulation, Proc. IEEE Virtual Real. Int. Symp. (1993) pp. 76–82

43.50 J.J. Abbott, P. Marayong, A.M. Okamura: Haptic virtual fixtures for robot-assisted manipulation, Proc. 12th Int. Symp. Robotics Res. (2007) pp. 49–64

43.51 D.J. Lee, P.Y. Li: Passive bilateral control and tool dynamics rendering for nonlinear mechanical teleoperators, IEEE Trans. Robotics **21**(5), 936–951 (2005)

43.52 M.J. Massimino, T.B. Sheridan, J.B. Roseborough: One handed tracking in six degrees of freedom, Proc. IEEE Int. Conf. Syst. Man Cybern. (1989) pp. 498–503

43.53 A. Ruesch, A.Y. Mersha, S. Stramigioli, R. Carloni: Kinetic scrolling-based position mapping for haptic teleoperation of unmanned aerial vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3116–3121

43.54 A. Casals, L. Munoz, J. Amat: Workspace deformation based teleoperation for the increase of movement precision, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 2824–2829

43.55 F. Conti, O. Khatib: Spanning large workspaces using small haptic devices, Proc. 1st Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst. (2005) pp. 183–188

43.56 R.W. Daniel, P.R. McAree: Fundamental limits of performance for force reflecting teleoperation, Int. J. Robotics Res. **17**(8), 811–830 (1998)

43.57 M.J. Massimino, T.B. Sheridan: Sensory substitution for force feedback in teleoperation, Presence Teleoperator Virtual Environ. **2**(4), 344–352 (1993)

43.58 D.A. Kontarinis, R.D. Howe: Tactile display of vibratory information in teleoperation and virtual environments, Presence Teleoperator Virtual Environ. **4**(4), 387–402 (1995)

43.59 D.A. Kontarinis, J.S. Son, W.J. Peine, R.D. Howe: A tactile shape sensing and display system for teleoperated manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 641–646

43.60 J.J. Gil, A. Avello, Á. Rubio, J. Flórez: Stability analysis of a 1 DOF haptic interface using the Routh–Hurwitz criterion, IEEE Trans. Control Syst. Technol. **12**(4), 583–588 (2004)

43.61 N. Hogan: Controlling impedance at the man/machine interface, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1626–1631

43.62 R.J. Adams, B. Hannaford: Stable haptic interaction with virtual environments,

Avidbots Ex1016
Page 1135 of 2154

IEEE Trans. Robotics Autom. **15**(3), 465–474 (1999)
43.63 K. Hashtrudi-Zaad, S.E. Salcudean: Analysis of control architectures for teleoperation systems with impedance/admittance master and slave manipulators, Int. J. Robotics Res. **20**(6), 419–445 (2001)
43.64 Y. Yokokohji, T. Yoshikawa: Bilateral control of master-slave manipulators for ideal kinesthetic coupling – Formulation and experiment, IEEE Trans. Robotics Autom. **10**(5), 605–620 (1994)
43.65 K.B. Fite, J.E. Speich, M. Goldfarb: Transparency and stability robustness in two-channel bilateral telemanipulation, ASME J. Dyn. Syst. Meas. Control **123**(3), 400–407 (2001)
43.66 S.E. Salcudean, M. Zhu, W.-H. Zhu, K. Hashtrudi-Zaad: Transparent bilateral teleoperation under position and rate control, Int. J. Robotics Res. **19**(12), 1185–1202 (2000)
43.67 W.R. Ferrell: Remote manipulation with transmission delay, IEEE Trans. Hum. Factors Electron. **6**, 24–32 (1965)
43.68 T.B. Sheridan: Space teleoperation through time delay: Review and prognosis, IEEE Trans. Robotics Autom. **9**(5), 592–606 (1993)
43.69 A. Eusebi, C. Melchiorri: Force reflecting telemanipulators with time-delay: Stability analysis and control design, IEEE Trans. Robotics Autom. **14**(4), 635–640 (1998)
43.70 W.S. Kim, B. Hannaford, A.K. Bejczy: Force-reflection and shared compliant control in operating telemanipulators with time delays, IEEE Trans. Robotics Autom. **8**(2), 176–185 (1992)
43.71 K. Hashtrudi-Zaad, S.E. Salcudean: Transparency in time-delayed systems and the effect of local force feedback for transparent teleoperation, IEEE Trans. Robotics Autom. **18**(1), 108–114 (2002)
43.72 R. Oboe, P. Fiorini: A design and control environment for internet-based telerobotics, Int. J. Robotics Res. **17**(4), 433–449 (1998)
43.73 S. Munir, W.J. Book: Control techniques and programming issues for time delayed internet based teleoperation, ASME J. Dyn. Syst. Meas. Control **125**(2), 205–214 (2003)
43.74 S. Hirche, M. Buss: Transparent data reduction in networked telepresence and teleaction systems. Part II: Time-delayed communication, Presence Teleoperator Virtual Environ. **16**(5), 532–542 (2007)
43.75 R.J. Anderson, M.W. Spong: Bilateral control of tele-operators with time delay, IEEE Trans. Autom. Control **34**(5), 494–501 (1989)
43.76 G. Niemeyer, J.-J.E. Slotine: Telemanipulation with time delays, Int. J. Robotics Res. **23**(9), 873–890 (2004)
43.77 S. Stramigioli, A. van der Schaft, B. Maschke, C. Melchiorri: Geometric scattering in robotic telemanipulation, IEEE Trans. Robotics Autom. **18**(4), 588–596 (2002)
43.78 N.A. Tanner, G. Niemeyer: High-frequency acceleration feedback in wave variable telerobotics, IEEE/ASME Trans. Mechatron. **11**(2), 119–127 (2006)
43.79 S. Munir, W.J. Book: Internet-based teleoperation using wave variables with prediction, IEEE/ASME Trans. Mechatron. **7**(2), 124–133 (2002)
43.80 D.J. Lee, M.W. Spong: Passive bilateral teleoperation with constant time delay, IEEE Trans. Robotics **22**(2), 269–281 (2006)
43.81 E. Nuno, L. Basanez, R. Ortega, M.W. Spong: Position tracking for non-linear teleoperators with variable time delay, Int. J. Robotics Res. **28**(7), 895–910 (2009)
43.82 K. Huang, D.J. Lee: Consensus-based peer-to-peer control architecture for multiuser haptic interaction over the internet, IEEE Trans. Robotics **29**(2), 417–431 (2013)
43.83 K. Huang, D.J. Lee: Hybrid pd-based control framework for passive bilateral teleoperation over the Internet, Proc. IFAC World Congr. (2011) pp. 1064–1069
43.84 B. Hannaford, J.H. Ryu: Time domain passivity control of haptic interfaces, IEEE Trans. Robotics Autom. **18**(1), 1–10 (2002)
43.85 J.-H. Ryu, C. Preusche, B. Hannaford, G. Hirzinger: Time domain passivity control with reference energy following, IEEE Trans. Control Syst. Technol. **13**(5), 737–742 (2005)
43.86 J. Artigas, C. Preusche, G. Hirzinger: Time domain passivity-based telepresence with time delay, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4205–4210
43.87 J. Ryu, C. Preusche: Stable bilateral control of teleoperators under time-varying communication delays: Time domain passivity approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 3508–3513
43.88 D.J. Lee, K. Huang: Passive-set-position-modulation framework for interactive robotic systems, IEEE Trans. Robotics Autom. **26**(2), 354–369 (2010)
43.89 J.P. Kim, J. Ryu: Robustly stable haptic interaction control using an energy-bounding algorithm, Int. J. Robotics Res. **29**(6), 666–679 (2010)
43.90 M.C.J. Franken, S. Stramigioli, S. Misra, S. Secchi, A. Macchelli: Bilateral telemanipulation with time delays: A two-layer approach combining passivity and transparency, IEEE Trans. Robotics **27**(4), 741–756 (2011)
43.91 V. Duindam, A. Macchelli, S. Stramigioli, H. Bruyninckx: *Modeling and Control of Complex Physical Systems* (Springer, Berlin, Heidelberg 2009)
43.92 S. Stramigioli: *Modeling and IPC Control of Interactive Mechanical Systems: A Coordinate-Free Approach*, Lecture Notes in Control and Information Sciences, Vol. 266 (Springer, London 2001)
43.93 S. Stramigioli, C. Secchi, A.J. Van der Schaft, C. Fantuzzi: Sampled Data Systems Passivity and Discrete Port-Hamiltonian Systems, IEEE Trans. Robotics **21**(4), 574–587 (2005)
43.94 D.J. Lee, O. Martinez-Palafox, M.W. Spong: Bilateral teleoperation of a wheeled mobile robot over delayed communication networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 3298–3303

Avidbots Ex1016
Page 1136 of 2154

43.95 N. Chopra, M.W. Spong, R. Lozano: Synchronization of bilateral teleoperators with time delay, Automatica **44**, 2142–2148 (2008)

43.96 J.-J.E. Slotine, W. Li: On the adaptive control of robot manipulators, Int. J. Robotics Res. **6**(3), 49–59 (1987)

43.97 D.J. Lee, D. Xu: Feedback *r*-passivity of lagrangian systems for mobile robot teleoperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 2118–2123

43.98 S. Stramigioli, R. Mahony, P. Corke: A novel approach to haptic tele-operation of aerial robot vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 5302–5308

43.99 D.J. Lee, A. Franchi, H.-I. Son, C. Ha, H.H. Bülthoff, P.R. Giordano: Semi-autonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles, IEEE/ASME Trans. Mech. **18**, 1334–1345 (2013)

43.100 H.I. Son, A. Franchi, L.L. Chuang, J. Kim, H.H. Bülthoff, P.R. Giordano: Human-centered design and evaluation of haptic cueing for teleoperation of multiple mobile robots, IEEE Trans. Cybern. **43**(2), 597–609 (2013)

43.101 D.J. Lee, M.W. Spong: Bilateral teleoperation of multiple cooperative robots over delayed communication networks: Theory, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 362–367

43.102 P.F. Hokayem, M.W. Spong: Bilateral teleoperation: An historical survey, Automatica **42**, 2035–2057 (2006)

43.103 A. Franchi, C. Secchi, H.I. Son, H.H. Bülthoff, P.R. Giordano: Bilateral teleoperation of groups of mobile robots with time-varying topology, IEEE Trans. Robotics **28**(5), 1019–1033 (2012)

43.104 G. Hwang, H. Hashimoto: Development of a human-robot-shared controlled teletweezing system, IEEE Trans. Control Sys. Technol. **15**(5), 960–966 (2007)

43.105 E.J. Rodriguez-Seda, J.J. Troy, C.A. Erignac, P. Murray, D.M. Stipanovic, M.W. Spong: Bilateral teleoperation of multiple mobile agents: Coordinated motion and collision avoidance, IEEE Trans. Control Sys. Technol. **18**(4), 984–992 (2010)

43.106 D.J. Lee, P.Y. Li: Passive decomposition of multiple mechanical systems under motion coordination requirements, IEEE Trans. Autom. Control **58**(1), 230–235 (2013)

43.107 P. Malysz, S. Sirouspour: Trilateral teleoperation control of kinematically redundant robotic manipulators, Int. J. Robotics Res. **30**(13), 1643–1664 (2011)

43.108 B. Khademian, K. Hashtrudi-Zaad: Shared control architectures for haptic training: performance and coupled stability analysis, Int. J. Robotics Res. **30**(13), 1627–1642 (2011)

43.109 M. Ferre, M. Buss, R. Aracil, C. Melchiorri, C. Balague (Eds.): *Advances in Telerobotics*, Springer Tracts in Advanced Robotics, Vol. 31 (Springer, Berlin, Heidelberg 2007)

Avidbots Ex1016
Page 1137 of 2154

Multimedia Contents

44. Networked Robots

Dezhen Song, Ken Goldberg, Nak-Young Chong

As of 2013, almost all robots have access to computer networks that offer extensive computing, memory, and other resources that can dramatically improve performance. The underlying enabling framework is the focus of this chapter: networked robots. Networked robots trace their origin to telerobots or remotely controlled robots. Telerobots are widely used to explore undersea terrains and outer space, to defuse bombs and to clean up hazardous waste. Until 1994, telerobots were accessible only to trained and trusted experts through dedicated communication channels. This chapter will describe relevant network technology, the history of networked robots as it evolves from teleoperation to cloud robotics, properties of networked robots, how to build a networked robot, example systems. Later in the chapter, we focus on the recent progress on cloud robotics, and topics for future research.

44.1 **Overview and Background** 1109
44.2 **A Brief History** 1110
44.2.1 Networked Teleoperation 1110
44.2.2 Cloud Robotics and Automation .. 1111
44.3 **Communications and Networking** 1112
44.3.1 The Internet 1113
44.3.2 Wired Communication Links 1113
44.3.3 Wireless Links 1114
44.3.4 Video and Audio Transmission Standards 1114
44.4 **Properties of Networked Robots** 1115
44.4.1 Overall Structure 1115
44.4.2 Building a Networked Robot System 1116
44.4.3 State-Command Presentation 1117
44.4.4 Command Execution/State Generation 1119
44.4.5 Virtual Fixtures 1119
44.4.6 Collaborative Control and Crowd Sourcing 1120
44.5 **Cloud Robotics** 1121
44.5.1 Big Data 1122
44.5.2 Cloud Computing 1122
44.5.3 Collective Robot Learning 1123
44.5.4 Open Source and Open Access 1123
44.5.5 Crowdsourcing and Call Centers .. 1124
44.6 **Conclusion and Future Directions** 1125
Video-References 1126
References 1126

44.1 Overview and Background

As illustrated in Fig. 44.1, the field of networked robots locates at the intersection between two exciting fields: robotics and networking. Similarly, teleoperation (Chap. 43) and multiple mobile robot systems (Chap. 51) also find their overlaps in the intersection. The primary concerns of the teleoperation are stability and time delay. Multiple mobile robot systems concerns coordination and planning of autonomous robots and sensors communicating over local networks. *The subfield of Networked Robots focuses on the robot system architectures, interfaces, hardware, software, and applications that use networks (primarily the Internet/Cloud).*

By 2012, several hundred networked robots have been developed and put online for public use. Many papers have been published describing these systems and a book on this subject by *Goldberg* and *Siegwart* is available [44.1]. Updated information about new research and an archive/survey of networked robots is available on the website of the IEEE technical committee on networked robots, which fosters research in this area [44.2].

Avidbots Ex1016
Page 1138 of 2154

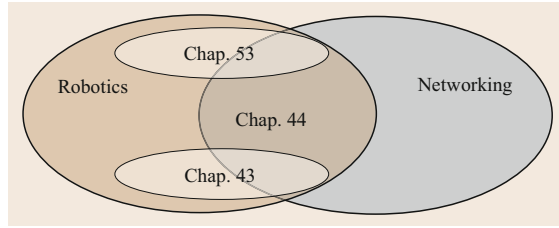


Fig. 44.1 Relationship between the subjects of networked robots (Chap. 44, this chapter), teleoperation (Chap. 43), and multiple mobile robot systems (Chap. 53) ►

The rest of the chapter is organized as follows: we first review the history and related work in Sect. 44.2. In Sect. 44.3, we review network and communication technology to provide necessary background for the following two main Sects. 44.4 and 44.5. Section 44.4 focuses on traditional networked robots while Sect. 44.5 summarize the new development in cloud robotics. Section 44.6, we conclude the chapter with recent applications and future directions.

44.2 A Brief History

Networked robots have their root in teleoperation systems, which started as remotely controlled devices. However, thanks to the recent evolution of the Internet and wireless networks, networked robots quickly expand their scope from the traditional master–slave teleopereation relationship to an integration of robots, human, agents, off-board sensors, databases, and clouds over the globe. To review the history of networked robots, we trace back to the root: remotely controlled devices.

44.2.1 Networked Teleoperation

Like many technologies, remotely controlled devices were first imagined in science fiction. In 1898, *Nicola Tesla* [44.3] demonstrated a radio-controlled boat in New York's Madison Square Garden. The first major experiments in teleoperation were motivated by the need to handle radioactive materials in the 1940s. *Goertz* and *Thompson* demonstrated one of the first bilateral simulators in the 1950s at the Argonne National Laboratory [44.4]. Remotely operated mechanisms have been designed for use in inhospitable environments such as undersea [44.5] and space exploration [44.6]. At General Electric, *Mosher* [44.7] developed a two-arm teleoperator with video cameras. Prosthetic hands were also applied to teleoperation [44.8]. More recently, teleoperation is being considered for medical diagnosis [44.9], manufacturing [44.10] and micromanipulation [44.11]. See Chap. 43 and the book from *Sheridan* [44.12] for excellent reviews on teleoperation and telerobotics research.

The concept of hypertext (linked references) was proposed by Vannevar Bush in 1945 and was made possible by the subsequent developments in computing and networking. In the early 1990s, Berners-Lee introduced the hypertext transmission protocol (HTTP). A group of students led by Marc Andreessen developed an open source version of the first graphical user interface, the *Mosaic* browser, and put it online in 1993. The first networked camera, the predecessor of today's *webcam*, went online in November 1993 [44.13]

Approximately nine months later, the first networked telerobot went online. The *Mercury Project* combined an IBM industrial robot arm with a digital camera and used the robot's air nozzle to allow remote users to excavate for buried artifacts in a sandbox [44.14, 15]. Working independently, a team led by *Taylor* and *Trevelyan* at the University of Western Australia demonstrated a remotely controlled six-axis telerobot in September 1994 [44.16, 17]. These early projects pioneered a new field of networked telerobots. See [44.18–26] for other examples.

Networked telerobots are a special case of *supervisory control* telerobots, as proposed by *Sheridan* and his colleagues [44.12]. Under supervisory control, a local computer plays an active role in closing the feedback loop. Most networked robotics are type (c) supervisory control systems (Fig. 44.2).

Although a majority of networked telerobotic systems consist of a single human operator and a single robot [44.27–34], *Chong* et al. [44.35] propose a useful taxonomy: single operator single robot (SOSR), single operator multiple robot (SOMR) [44.36, 37], multiple operator single robot (MOSR), and multiple operator multiple robot (MOMR) [44.38, 39] (VIDEO 81, VIDEO 84). These frameworks greatly extend system architecture of networked robots. In fact, human operators can often be replaced with autonomous agents, off-board sensors, expert systems, and programmed logics, as demonstrated by *Xu* and *Song* [44.40] and *Sanders* et al. [44.41]. The extended networked connectivity also allows us to employ techniques such as crowd sourcing and collaborative control for demanding applications such as nature observation and environment monitoring [44.42, 43]. Hence, networked teler-

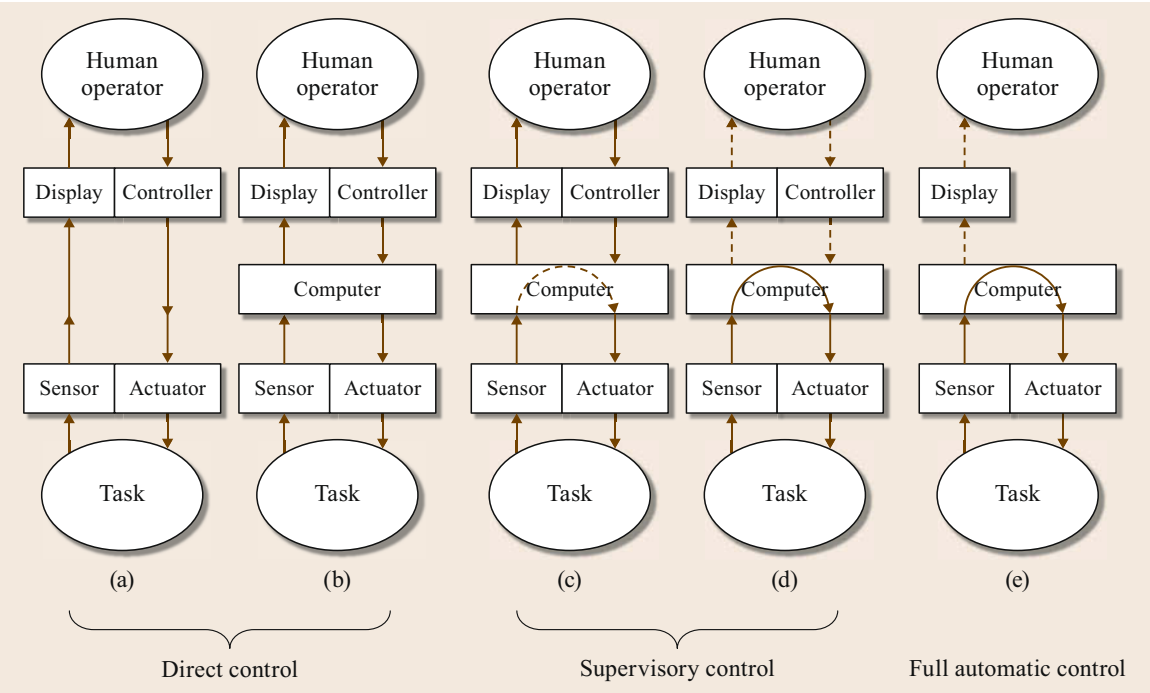


Fig. 44.2 A spectrum of teleoperation control modes adapted from *Sheridan*'s text (after [44.12]). We label them (a–e), in order of increasing robot autonomy. At the far left would be a mechanical linkage where the human directly operates the robot from another room through sliding mechanical bars, and on the far right is the system where the human role is limited to observation/monitoring. In (c–e), the dashed lines indicated that communication may be intermittent

obots fully evolute into networked robots: an integration of robots, humans [44.44], computing power, off-board sensing, and databases over the Internet.

The last 18 years (1994–2012) witnessed the extensive development in networked robots. New systems, new experiments, and new applications go well beyond traditional fields such as defense, space, and nuclear material handing [44.12] that motivated teleoperation in early 1950s. As the Internet introduces universal access to every corner of life, the impact of networked robots becomes broader and deeper in modern society. Recent applications range from education, industry, commercial, health care, geology, environmental monitoring, to entertainment, and arts.

Networked robots provide a new medium for people to interact with remote environment. A networked robot can provide more interactivity beyond what a normal videoconferencing system. The physical robot not only represents the remote person but also transmits multimodal feedback to the person, which is often referred as *telepresence* in the literature [44.30]. *Paulos* et al.'s Personal ROving Presence (PRoP) robot [44.45], *Jouppi* and *Thomas*' Surrogate robot [44.30], *Takayama* et al.'s Texai [44.46], and *Lazewatsky* and *Smart*'s inexpensive platform [44.47] are representative work.

Networked robots have great potential for education and training. In fact, one of the earliest networked telerobot systems [44.48] originates from the idea of a remote laboratory. Networked telerobots provide universal access to the general public, who may have little to no knowledge of robots, with opportunities to understand, learn, and operate robots, which were expensive scientific equipment limited to universities and large corporate laboratories before. Built on networked telerobots, online remote laboratories [44.49, 50] greatly improves distance learning by providing an interactive experience. For example, teleoperated telescopes help students to understand astronomy [44.51]. Teleoperated microscope [44.52] helps student to observe micro-organisms. The Tele-Actor project [44.53] allows a group of students to remotely control a human tele-actor to visit environments that are normally not accessible to them such as clean-room environments for semiconductor manufactory facility and deoxyribonucleic acid (DNA) analysis laboratories.

44.2.2 Cloud Robotics and Automation

Recent development of cloud computing provide new means and platform for networked robots. In

2010, *Kuffner* at Google introduced the term *Cloud robotics* [44.54] to describe a new approach to robotics that takes advantage of the Internet as a resource for massively parallel computation and real-time sharing of vast data resources. The Google autonomous driving project exemplifies this approach: the system indexes maps and images that are collected and updated by satellite, Streetview, and crowdsourcing from the network to facilitate accurate localization. Another example is Kiva Systems new approach to warehouse automation and logistics using large numbers of mobile platforms to move pallets using a local network to coordinate planforms and update tracking data. These are just two new projects that build on resources from the Cloud. Steve Cousins of Willow Garage aptly summarized the idea: *No robot is an island*. Cloud robotics recognizes the wide availability of networking, incorporates elements of open-source, open-access, and crowdsourcing to greatly extend earlier concepts of *Online Robots* [44.1] and *Networked Robots* [44.55, 56].

The Cloud has been used as a metaphor for the Internet since the inception of the World Wide Web in the early 1990s. As of 2012, researchers are pursuing a number of cloud robotics and automation projects [44.57, 58]. New resources range from software architectures [44.59–62] to computing resources [44.63]. The RoboEarth project [44.64] aims to develop [44.65]:

> *a World Wide Web for robots: a giant network and database repository where robots can share information and learn from each other about their behavior and their environment.*

Cloud robotics and automation is related to concepts of the *Internet of Things* [44.66] and the *Industrial Internet*, which envision how radio-frequency identification (RFID) and inexpensive processors can be incorporated into a vast array of objects from inventory items to household appliances to allow them to communicate and share information.

44.3 Communications and Networking

Below is a short review of relevant terminologies and technologies on networking. For details, see the texts by [44.67].

A communication network includes three elements: *links*, *routers/switchers*, and *hosts*. Links refer to the physical medium that carry bits from one place to another. Examples of links include copper or fiber-optic cables and wireless (radio frequency or infrared) channels. Switches and routers are hubs that direct digital information between links. Hosts are communication end points such as browsers, computers, and robots.

Networks can be based in one physical area (local-area network, or LAN), or distributed over wide distances (wide-area network, or WAN). Access control is a fundamental problem in networking. Among a variety of methods, the *ethernet* protocol is the most popular. Ethernet provides a broadcast-capable multi-access LAN. It adopts a carrier-sense multiple-access (CSMA) strategy to address the multiple-access problem. Defined in the IEEE 802.x standard, CSMA allows each host to send information over the link at any time. Therefore, collisions may happen between two or more simultaneous transmission requests. Collisions can be detected either by directly sensing the voltage in the case of wired networks, which is referred to as collision detection (CSMA/CD), or by checking the time-out of an anticipated acknowledgement in wireless networks, which is referred to as collision avoidance (CSMA/CA). If a collision is detected, both/all senders randomly back off a short period of time before retransmitting. CSMA has a number of important properties: (1) it is a completely decentralized approach, (2) it does not need clock synchronization over the entire network, and (3) it is very easy to implement. However, the disadvantages of CSMA are: (1) the efficiency of the network is not very high and (2) the transmission delay can change drastically.

As mentioned previously, LANs are interconnected with each other via routers/switchers. The information transmitted is in packet format. A packet is a string of bits and usually contains the source address, the destination address, content bits, and a checksum. Routers/switchers distribute packets according to their routing table. Routers/switchers have no memory of packets, which ensures scalability of the network. Packets are usually routed according to a first-in first-out (FIFO) rule, which is independent of the application. The packet formats and addresses are independent of the host technology, which ensures extensibility. This routing mechanism is referred to as packet switching in the networking literature. It is quite different from a traditional telephone network, which is referred to as circuit switching. A telephone network is designed to guarantee a dedicated circuit between a sender and a receiver once a phone call is established. The dedicated circuitry ensures communication quality. However, it requires a large number of circuits to ensure the quality of service (QOS), which leads to poor utilization of the overall network. A packet-switching network cannot guarantee dedicated bandwidth for each indi-

vidual pair of transmissions, but it improves overall resource utilization. The Internet, which is the most popular communication media and the infrastructure of networked telerobots, is a packet-switching network.

44.3.1 The Internet

The creation of the Internet can be traced back to US Department of Defense's (DOD) APRA NET network in the 1960s. There are two features of the APRA NET network that enabled the successful evolution of the Internet. One feature is the ability for information (packets) to be rerouted around failures. Originally, this was designed to ensure communication in the event of a nuclear war. Interestingly, this dynamic routing capability also allows the topology of the Internet to grow easily. The second important feature is the ability for heterogeneous networks to interconnect with one another. Heterogeneous networks, such as X.25, G.701, ethernet, can all connect to the Internet as long as they can implement the Internet protocol (IP). The IP is media, operating system (OS), and data rate independent. This flexible design allows a variety of applications and hosts to connect to the Internet as long as they can generate and understand IP.

Figure 44.3 illustrates a four-layer model of the protocols used in the Internet. On the top of the IP, we have two primary transport layer protocols: the transmission control protocol (TCP) and the user data protocol (UDP). TCP is an end-to-end transmission control protocol. It manages packet ordering, error control, rate control, and flow control based on packet round-trip time. TCP guarantees the arrival of each packet. However, excessive retransmission of TCP in a congested network may introduce undesirable time delays in a networked telerobotic system. UDP behaves differently; it is a broadcast-capable protocol and does not have a retransmission mechanism. Users must take care of error control and rate control themselves. UDP has a lot less overhead compared to TCP. UDP packets are transmitted at the sender's preset rate and the rate is changed based on the congestion of a network. UDP has great potential, but it is often blocked by firewalls because of a lack of a rate control mechanism. It is also worth mentioning that the widely accepted term TCP/IP refers to the family of protocols that build on IP, TCP, and UDP.

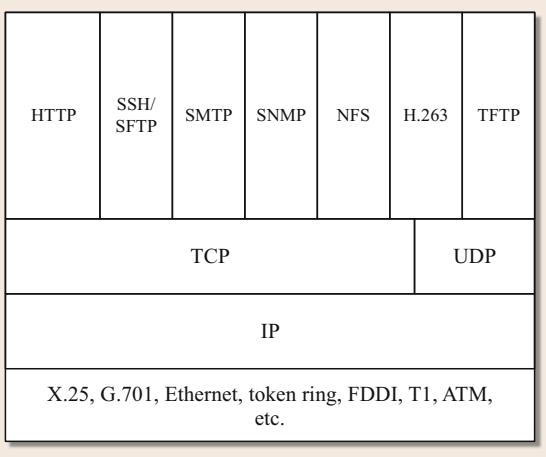


Fig. 44.3 A four-layer model of Internet protocols (after [44.67])

In the application layer of the Internet protocols, the HTTP is one of the most important protocols. HTTP is the protocol for the World Wide Web (WWW). It allows the sharing of multimedia information among heterogeneous hosts and OSs including text, image, audio, and video. The protocol has significantly contributed to the boom of the Internet. It also changes the traditional client/server (C/S) communication architecture to a browser/server (B/S) architecture. A typical configuration of the B/S architecture consists of a web server and clients with web browsers. The web server projects the contents in hypertext markup language (HTML) format or its variants, which is transmitted over the Internet using HTTP. User inputs can be acquired using the common gateway interface (CGI) or other variants. The B/S architecture is the most accessible because no specialized software is needed at the client end.

44.3.2 Wired Communication Links

Even during peak usage, the network backbones of the Internet often run at less than 30% of their overall capacity. The average backbone utilization is around 15–20%. The primary speed limitation for the Internet

Table 44.1 *Last-mile* Internet speed by wired connection type (if not specified, the downstream transmission and the upstream transmission share the same bandwidth)

Types	Bits per second
Dialup modem (V.92)	Up to 56 K
Integrated services digital network (ISDN)	64–160 K for BRI, up to 2048 K for PRI
High data rate digital subscriber line (HDSL)	Up to 2.3 M duplex on two twisted-pair lines
Asymmetric digital subscriber line (ADSL)	1.544–24.0 M downstream, 0.5–3.3 M upstream
Cable modem	2–400 M downstream, 0.4–108 M upstream
Fiber to the home (FTTH)	0.005–1 G downstream, 0.002–1 G upstream
Direct Internet II node	1.0–10.0 G

Table 44.2 Survey of wireless technologies in terms of bit rate and range

Types	Bit rate (bps)	Band (Hz)	Range (m)
Zigbee (802.15.4)	20–250 K	868–915 M/2.4 G	50
Bluetooth	732 K–3.0 M	2.4 G	100
3G HSPA	400 K–14.0 M	$\leq$ 3.5 G	N/A
HSPA+	5.76 M–44.0 M	$\leq$ 3.5 G	N/A
LTE	10 M–300 M	$\leq$ 3.5 G	N/A
WiFi (802.11a,b,g,n)	11–600 M	2.4 G/5 G	100

is the *last mile*, the link between clients and their local Internet service providers (ISP).

Table 44.1 lists typical bit rates for different connection types. It is interesting to note the asymmetric speeds in many cases, where upstream bit rate (from the client to the Internet), are far slower than downstream bit rates (from the Internet to the client). These asymmetries introduce complexity into the network model for teleoperation. Since the speed difference between the slowest modem link and the fastest Internet II node is over 10 000, designers of a networked telerobotic system should anticipate a large variance of communication speeds.

44.3.3 Wireless Links

Table 44.2 compares the speed, band, and range of wireless standards as of 2012. Increasing bit rate and communication range requires increasing power. The amount of radio frequency (RF) transmission power required over a distance d is proportional to d^k, where $2 \leq k \leq 4$ depending on the antenna type. In Table 44.2, Bluetooth and Zigbee are typical low-power transmission standards that are good for short distances. HSPA+ and LTE are commercially marketed as the 4G cellphone network.

By providing high-speed connectivity at low cost, WiFi is the most popular wireless standard in 2012. Its range is approximate 100 m line of sight and the WiFi wireless network usually consists of small-scale interconnected access points. The coverage range usually limits these networks to an office building, home, and other indoor environments. WiFi is a good option for indoor mobile robots and human operators. If the robot needs to navigate in the outdoor environment, the 3G or 4G cellphone network can provide the best coverage available. Although obvious overlap exists among wireless standards in coverage and bandwidth, there are two import issues that have not been covered by Table 44.2. One is mobility. We know that, if an RF source or receiver is moving, the corresponding Doppler effect causes a frequency shift, which could cause problems in communication. WiFi is not designed for fast-moving hosts. 3G HSPA cellphone allows the host to move at a vehicle speed under 120 km/h. However, LTE allows the host to move at a speed of 350 or 500 km/h, which even works for high-speed trains.

Long range wireless links often suffer from latency problem, which may drastically decreases system performer as discussed in Chap. 43. One may notice that we did not list satellite wireless in Table 44.2 because the long latency (0.5–1.7 s) and high price makes it difficult to be useful for robots. The large antenna size and high power consumption rate also limits its usage in mobile robots. In fact, the best option for long range wireless is LTE. LTE is designed with a transmission latency of less that 4 ms whereas 3G HSPA cellphone networks have a variable latency of 10–500 ms.

44.3.4 Video and Audio Transmission Standards

In networked robots systems, the representation of the remote environment is often needed to be delivered to online users in video and audio format. To deliver video and audio over the Internet, raw video and audio data from camera optical sensor and microphone must be compressed according to different video and audio compression standards to fit in the limited network bandwidth. Due to lack of bandwidth and computing power to encode streaming video, most early systems only transmit periodic snapshots of the remote scene in JPEG format at limited frame rate, i. e., 1–2 frames per second or less. Audio was rarely considered in the early system design. The rudimentary video delivery methods in the early system were mostly implemented using HTML and JavaScript to reload the JPEG periodically.

Today, the expansion of HTML standards allow web browsers to employ plug-ins as the client end of streaming video. HTML5 even natively supports video decoding. Therefore, the server end of recent systems often employs streaming server software, such as Adobe Flash Media Encoder, Apple Quick Time Streaming Server, Oracle Java Media Framework, Helix Media Delivery Platform, Microsoft DirectX, SkypeKit, etc., to encode and deliver video. These streaming video sever packages often provide easy-to-use software development kit (SDK) to facilitate system integration.

It is worth noting that these different software packages are just different implementations of video/audio

Table 44.3 A comparison of existing videostreaming standards for the same resolution under the same fixed bandwidth (FMBT represents buffering time settings that would not significantly decrease compression ratio or video quality.)

Standards	Feasible minimum buffering time (FMBT)	Framerate
MJPEG	Zero (< 10 ms)	Low
MPEG2	Variable (i. e., 50 ms – video length), 2–10 s are common	Moderate
H.263+	< 300 ms	High
H.264/ MPEG4-AVC	Zero (< 10 ms)	Highest

streaming protocols. Not every protocol is suitable for networked robots. Some protocols are designed to deliver video on demand while others are designed for live streaming for videoconferencing purposes. Networked robots use real-time video as feedback information, which imposes strict requirements in latency and bandwidth similar to those of videoconferencing. One way latency of more than 150 ms can significantly degrade telepresence and hence the performance of the human operator.

Latency is often caused by bandwidth and video encoding/decoding time. Since audio data amount is negligible when comparing to that video data. We will focus the discussion on video compression standards. There is always a tradeoff between frame rate and resolution for a given bandwidth. There is also a tradeoff between compression ratio and computation time for a given central processing unit (CPU). The computation time includes both CPU time and data-buffering time at both client and server ends. Video encoding is a very computationally intensive task. A long computation period introduces latency and significantly impair the system performance. It is possible to use hardware to cut down the computation time but not the data-buffering time, which are controlled by the video encoder.

There are many standards and protocols available but most of them are just variations of MJPEG, MPEG2, H.263, and MPEG4/AVC/H.264. We compare those standards in Table 44.3. Note that the comparison is qualitative and may not be the most accurate due to the fact that each video encoding standard has many parameters that affect the overall buffering time.

From networked robot point of view, the buffering time determines the latency and the frame rate determines the responsiveness of the system. An ideal videostream should have both high frame rate and low buffering time. But if both cannot be achieved at the same time, low latency is preferred. From Table 44.3, H.264/MPEG4-AVC clearly outperforms other competitors and is the most popular video compression method.

44.4 Properties of Networked Robots

Networked robots have the following properties:

- The physical world is affected by a device that is locally controlled by a network *server*, which connects to the Internet to communicate with remote human users, databases, agents, and off-board sensors, which are referred to as *clients* of the system.
- Human decision making capability is often an integral part of the system. If so, humans often access the robot via web browsers, such as Internet Explorer or Firefox, or apps in mobile device. As of 2012, the standard protocol for network browsers is the hypertext transfer protocol (HTTP), a stateless transmission protocol.
- Most networked robots are continuously accessible (online), 24 h a day, 7 days a week.
- Networks may be unreliable or have different speed for clients with different connections.
- Since hundreds of millions of people now have access to the Internet, mechanisms are needed to handle client authentication and contention. System security and privacy of users are important in the networked robots.
- Input and output for human users for networked robots are usually achieved with the standard computer screen, mouse, and keyboard.
- Clients may be inexperienced or malicious, so online tutorials and safeguards are generally required.
- Additional sensing, databases, and computing resources may be available over the network.

44.4.1 Overall Structure

As defined by *Mason*, *Peshkin*, and others [44.68, 69], in *quasistatic* robot systems, accelerations, and inertial forces are negligible compared to dissipative forces. In quasistatic robot systems, motions are often modeled as transitions between discrete atomic *configurations*.

We adopt a similar terminology for networked telerobots. In quasistatic telerobotics (QT), robot dynamics and stability are handled locally. After each atomic motion, a new state report is presented to the remote user,

who sends back an atomic command. The atomic state describes the status of the robot and its corresponding environment. Atomic commands refer to human directives, which are desired robotic actions.

Several issues arise:

- *State-command presentation*: How should state and available commands be presented to remote human operators using the two-dimensional (2-D) screen display?
- *Command execution/state generation*: How should commands be executed locally to ensure that the desired state is achieved and maintained by the robot?
- *Command coordination*: How should commands be resolved when there are multiple human operators and/or agents? How to synchronize and aggregate commands issued by users/agents with different network connectivity, background, responsiveness, error rate, etc., to achieve the best possible system performance?
- *Virtual Fixture: Error prevention and state correction*: How should the system prevent the wrong commands that may lead the robot to collision or other undesirable states?

Before we detail these issues, let us walk through how to build a minimum networked robot system.

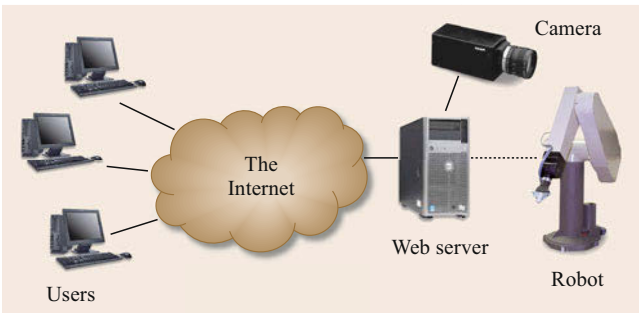


Fig. 44.4 Typical system architecture for a networked telerobot

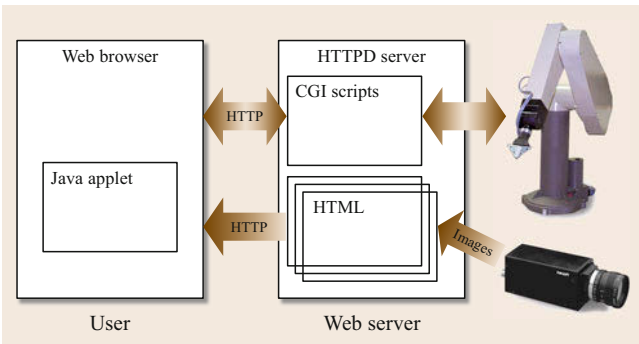


Fig. 44.5 A sample software architecture of a networked telerobot

A reader can follow the below example to build his/her owner networked robot system as well as understand challenges in the issues.

44.4.2 Building a Networked Robot System

This minimal system is a networked telerobotic system which allows a group of users to access a robot via web browsers. As illustrated in Fig. 44.4, a typical or minimal networked telerobotic system typically includes three components:

- *Users*: Anyone with an Internet connection and a web browser or equivalent apps that understand HTTP.
- *Web server*: A computer running a web server software.
- *Robot*: A robot manipulator, a mobile robot, or any device that can modify or affect its environment.

Users access the system via their web browsers. Any web browser that is compatible with W3C's HTML standard can access a web server. In 2012, the most popular web browsers are Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Apple Safari, and Opera. New browsers and updated versions with new features are introduced periodically. All of these popular browsers issue the corresponding mobile apps to support mobile devices such as Apple iPads, Apple iPhones, and Google Andriod-based Tablets and smart phones.

A web server is a computer that responds to HTTP requests over the Internet. Depending upon the operating system of the web server, popular server software packages include Apache and Microsoft Internet Information Services (IIS). Most servers can be freely downloaded from the Internet.

To develop a networked telerobot, one needs a basic knowledge of developing, configuring, and maintaining web servers. As illustrated in Fig. 44.5, the development requires knowledge of HTML and at least one local programming languages such as C, C#, CGI, Javascript, Perl, PHP, .Net, or Java.

It is important to consider compatibility with the variety of browsers. Although HTML is designed to be compatible with all browsers, there are exceptions. For example, Javascript, which is the embedded scripting language of web browsers, is not completely compatible between Internet Explorer and Firefox. One also needs to master the common HTML components such as forms that are used to accept user inputs, frames that are used to divide the interface into different functional regions, etc. An introduction to HTML can be found in [44.70].

User commands are usually processed by the web server using CGI, the common gateway interface. Most sophisticated methods such as PHP, Java Server Pages (JSP), and socket-based system programming can also be used. CGI is invoked by the HTTP server when the CGI script is referred in the uniform resource locator (URL). The CGI program then interprets the inputs, which is often the next robot motion command, and sends commands to the robot via a local communication channel. CGI scripts can be written in almost any programming language. The most popular ones are Perl and C.

A simple networked telerobotic system can be constructed using only HTML and CGI. However, if the robot requires a sophisticated control interface, advanced plug-ins such as Java Applet, Silver Light, or Flash, are recommended. These plug-ins run inside the web browser on the client's computer. Information about these plug-ins can be found at home pages of Oracle, Microsoft, and Adobe, respectively. Java applet is highly recommended because it is the most widely supported by different browsers. Recently, the fast adoption of HTML5 also provide a new long term solution to solve the compatibility issue.

Most telerobotic systems also collect user data and robot data. Therefore, database design and data processing program are also needed. The most common used databases include MySQL and PostgresSQL. Both are open-source databases and support a variety of platforms and operation systems. Since a networked telerobotic system is online 24 h a day, reliability is also an important consideration in system design. Website security is critical. Other common auxiliary developments include online documentation, online manual, and user feedback collection.

It is not difficult to expand this minimal networked telerobotic system into a full-fledged networked robot system. For example, some users can be replaced by agents that runs 24 h a day and 7 days a week to monitor system states and co-perform tasks with humans or take over the system when nobody is online. These agents can be implemented using cloud computing. Such extensions are usually based on the need of the task.

44.4.3 State-Command Presentation

To generate a correct and high-quality command depends on how effectively the human operator understands the state feedback. The state-command presentation contains three subproblems: the 2-D representation of the true robot state (state display), the assistance provided by the interface to generate new commands (spatial reasoning), and the input mechanism.

State Displays

Unlike traditional point-to-point teleoperation, where specialized training and equipment are available to operators, networked telerobots offer wide access to the general public. Designers cannot assume that operators

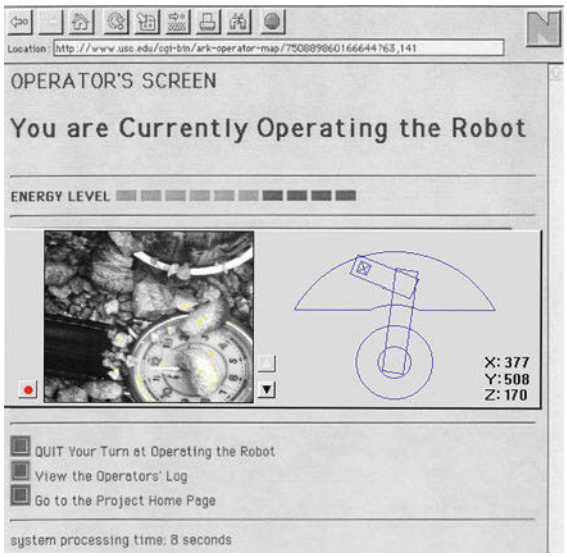


Fig. 44.6 Browser's view of the first networked telerobot interface (after [44.55]). The schematic at *lower right* gives an overhead view of position of the four-axis robot arm (with the camera at the end marked with X), and the image at the *lower left* indicates the current view of the camera. The *small button marked with a dot at the left* directs a 1 s burst of compressed air into the sand below the camera. The Mercury Project was online from August 1994 to March 1995

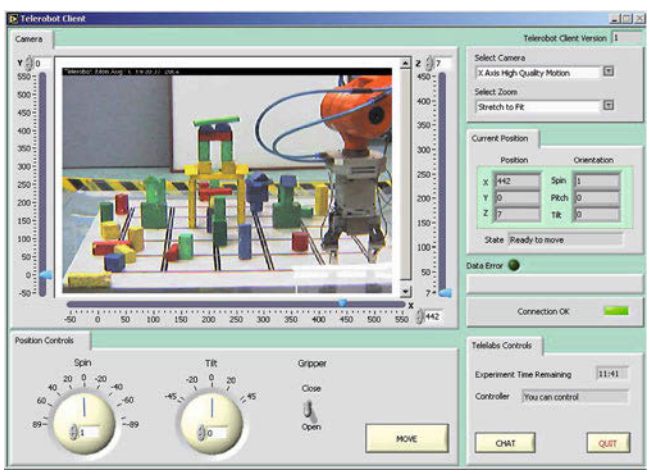


Fig. 44.7 Browser interface to the Australian networked telerobot which was a six-axis arm that could pick up and move blocks (after [44.17])

have any prior experience with robots. As illustrated in Fig.44.6, networked telerobotic systems must display the robot state on a 2-D screen display.

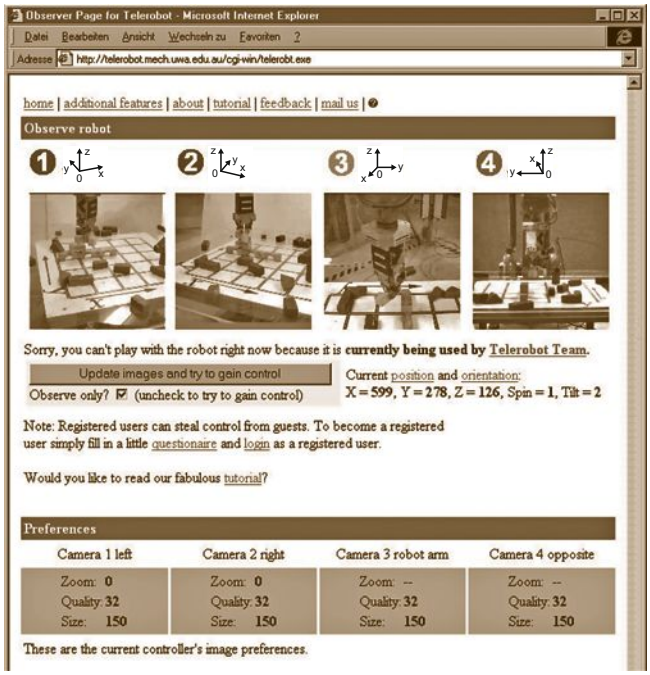


Fig. 44.8 Use of a multicamera system for multiviewpoint state feedback (after [44.71])

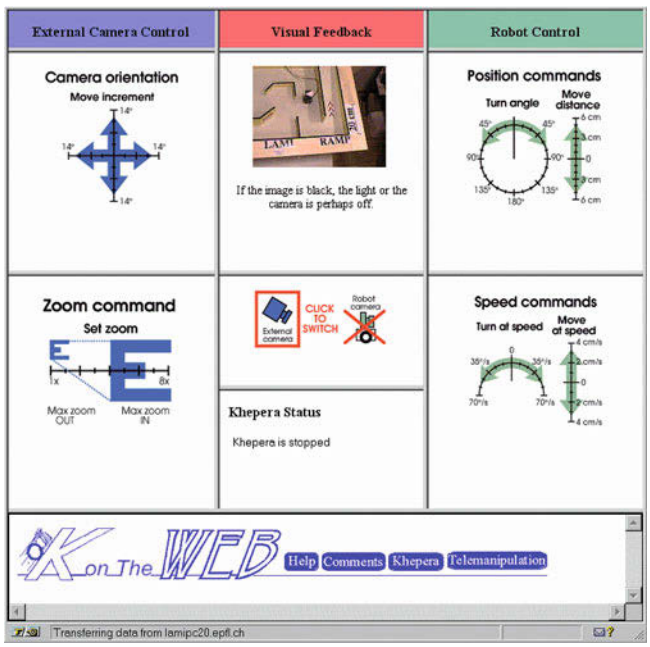


Fig. 44.9 Camera control and mobile robot control in Patrick Saucy and Francesco Mondada's Khep on the web project

The states of the teleoperated robot are often characterized in either world coordinates or robot joint configuration, which are either displayed in numerical format or through a graphical representation. Figure 44.6 lists robot *XYZ* coordinates on the interface and draws a simple 2-D projection to indicate joint configurations. Figure 44.7 illustrates another example of teleoperation interface that was developed by *Taylor* and *Trevelyan* [44.48]. In this interface, *XYZ* coordinates are presented in a sliding bar near the video window.

The state of the robot is usually displayed in a 2-D view as shown in Figs. 44.6 and 44.7. In some systems, multiple cameras can help the human operator to understand the spatial relationship between the robot and the objects in the surrounding environment. Figure 44.8 shows an example with four distinct camera views for a six-degree-of-freedom industrial robot.

Figure 44.9 demonstrate an interface with a pan–tilt–zoom robotic camera. The interface in Fig. 44.9 is designed for a mobile robot.

More sophisticated spatial reasoning can eliminate the need for humans to provide low-level control by automatically generating a sequence of commands after it receives task-level commands from the human operator. This is particularly important when the robotic system is highly dynamic and requires a very fast response. In this case, it is impossible to ask the human to generate intermediate steps in the robot control; for example, *Belousov* et al. adopt a shared autonomy model to direct a robot to capture a moving rod [44.28] as shown in Fig. 44.10. *Fong* and *Thorpe* [44.72] summarize vehicle teleoperation systems that utilize these supervisory control techniques. *Su* and *Luo* developed an incremental algorithm for better translation of the intention and motion of operators into remote robot action commands [44.33].

The fast development of sensing and display technology makes it possible to visualize robot and environment states in three-dimensional (3-D) displays or generate synthetic eco-centric views (a.k.a. third person views) (VIDEO 82(. To achieve that, it often requires the robot to be equipped with multiple cameras and laser range finders to quickly reconstruct the remote environment [44.73, 74]. Sometimes, the reconstructed sensory information can be superimposed on priorly known 3-D information to form an augmented reality. This kind of display can drastically increase telepresence and performance.

Human Operator Input

Most networked telerobotic systems only rely on mouses and keyboards for input. The design problem is what to click on in the interface. Given the fact that user commands can be quite different, we need to adopt

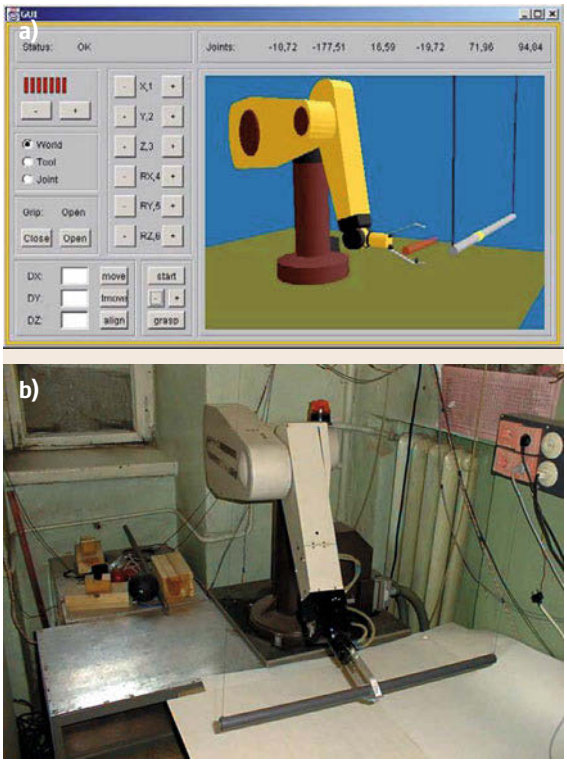


Fig.44.10a,b A web-based teleoperation system that allows a robot to capture a fast-moving rod. **(a)** User interface, and **(b)** system setup (after [44.28])

an appropriate interface for inputs; for example, inputs could be Cartesian *XYZ* coordinates in world coordinate system or robot configurations in angular joint configurations.

For angular inputs, it is often suggested to use a round dial as a control interface, as illustrated in bottom left of Fig. 44.7 and the right-hand side of Fig. 44.9. For linear motion in Cartesian coordinate, arrows operated by either mouse clicks or the keyboard are often suggested. Position and speed controls are often needed, as illustrated in Fig. 44.9. Speed control is usually controlled by mouse clicks on a linear progress bar for translation and a dial for rotation.

The most common control type is the position control. The most straightforward way is to click on the video image directly. To implement the function, the software needs to translate the 2-D click inputs into three-dimensional (3-D) world coordinates. To simplify the problem, the system designer usually assumes that the clicked position is on a fixed plane; for example, a mouse click on the interface of Fig. 44.6 assumes the robot moves on the *X*–*Y* plane. The combination of a mouse click on the image can also allow abstract task-level command. The example in Fig. 44.12 uses mouse clicks to place votes on an image to generate a command that directs a robot to pick up a test agent at the task level.

44.4.4 Command Execution/State Generation

When a robot receives a command, it executes the command and a new state is generated and transmitted back to the human operator. However, commands may not arrive in time or may get lost in transmission. Also, because users are often inexperienced, their commands may contain errors. Over the limited communication channel, it is impossible to ask the human to control the manipulator directly. Computer vision, laser range finder, local intelligence, and augmented-reality-based displays [44.74] are required to assist the human operator.

Belousov et al. demonstrated a system that allowed a web user to capture a fast rod that is thrown at a robot manipulator [44.28]. The rod is on bifilar suspension, performing complicated oscillations. Belousov et al. designed a shared-autonomy control to implement the capture. First, an operator chooses the desired point for capture on the rod and the capture instant using a 3-D online virtual model of the robot and the rod. Then, the capturing operation is performed automatically using a motion prediction algorithm that is based on the rod's motion model and two orthogonal camera inputs, which perceive the rod's position locally in real time.

This *shared autonomy* approach is often required when the task execution requires much faster response than the Internet can allow. Human commands have to remain at the task level instead of directing the movements of every actuators. The root of this approach can be traced back to the *Tele-Autonomous* concept proposed by *Conway* et al. [44.75] in 1990. In this paper, two important notions including time clutch and position clutches are introduced to illustrate the shared autonomy approach. The time clutch disengages the time synchronization between the human operator and the robot. The human operator verifies his/her commands on a predictive display before sending a set of verified commands to remote robots. The robot can then optimize the intermediate trajectory proposed by the human operator and disengage the position correspondence, which is referred to as the position clutch. Recent work [44.76] uses the similar idea to guide load-haul-dump vehicles in the underground mines by combining human inputs with tunnel following behavior.

44.4.5 Virtual Fixtures

Due to time delay, lack of background, and possible malicious behavior, human errors are inevitably introduced

to system from time to time. Erroneous states may be generated from the incorrect commands. If unchecked, robots or objects in the environment may be damaged. Sometimes, users may have good intention but are not able to generate accurate commands to control the robot remotely. For example, it is hard to generate a set of commands to direct a mobile robot to move along the wall and maintain a distance of 1 m to the wall at the same time.

Virtual fixtures are designed to cope with these challenges in teleoperation tasks. Proposed by *Rosenberg* [44.77], virtual fixtures are defined as an overlay of abstract sensory information on a robot workspace in order to improve the telepresence in a telemanipulation task. To further explain the definition, Rosenberg uses a ruler as an example. It is very difficult for a human to draw a straight line using bare hands. However, if a ruler, which is a physical fixture, is provided, then the task becomes easy. Similar to a physical fixture, a virtual fixture is designed to guide robot motion through some fictitious boundaries or force fields, such as virtual tubes or surface, generated according to sensory data. The virtual fixtures are often implemented using control laws [44.78, 79] based on a *virtual contact* model.

Virtual fixtures serve for two main purposes: avoiding operation mistakes and guide robots along the designable trajectories. This is also a type of shared autonomy that is similar to that in Sect. 44.4.4 where both the robot and the human share control in the system. Chapter 43 details the shared control scheme. It is worth noting that virtual fixtures should be visualized in the display to help operators understand the robot state to maintain situation awareness. This actually turns the display to *augmented reality* [44.80].

44.4.6 Collaborative Control and Crowd Sourcing

When more than one human is sharing control of the device, command coordination is needed. According to [44.81], multiple human operators can reduce the chance of errors, cope with malicious inputs, utilize operators' different expertise, and train new operators. In [44.82, 83], a collaboratively controlled networked robot is defined as a telerobot simultaneously controlled by many participants, where input from each participant is combined to generate a single control stream.

When group inputs are in the form of direction vectors, averaging can be used as an aggregation mechanism [44.84]. When decisions are distinct choices or at the abstract task level, voting is a better choice [44.53]. As illustrated in Fig. 44.11, *Goldberg* and *Song* (VIDEO 83) develop the Tele-Actor system using spatial dynamic voting. The Tele-Actor is a human equipped with an audio/video device and controlled by a group of online users. Users indicate their intensions by positioning their votes on a 320×320 pixel voting image during the voting interval. Votes are collected at the server and used to determine the Tele-Actor's next action based on the most requested region on the voting image [44.85].

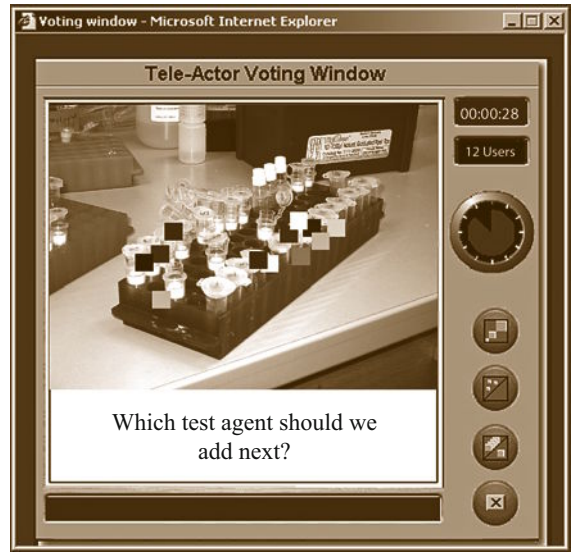


Fig. 44.11 Spatial dynamic voting interface for the Tele-Actor system: the spatial dynamic voting (SDV) interface as viewed by each user. In the remote environment, the Tele-Actor takes images with a digital camera, which are transmitted over the network and displayed to all participants with a relevant question. With a mouse click, each user places a color-coded marker (a *votel* or voting element) on the image. Users view the position of all votels and can change their votel positions based on the group's response. Votel positions are then processed to identify a *consensus region* in the voting image that is sent back to the Tele-Actor. In this manner, the group collaborates to guide the actions of the Tele-Actor (after [44.53])

Another approach to collaboratively control a networked robot is the employ a optimization framework. *Song* et al. [44.86, 87] developed a collaboratively controlled camera that allowed many clients to share control of its camera parameters, as illustrated in Fig. 44.12. Users indicate the area they want to view by drawing rectangles on a panoramic image. The algorithm computes an optimal camera frame with respect to the user satisfaction function, which is defined as the frame selection problem [44.88, 89].

Recent work by *Xu* et al. [44.40, 90] further the optimization framework to *p*-frames that allows multiple cameras to be controlled and coordinated whereas

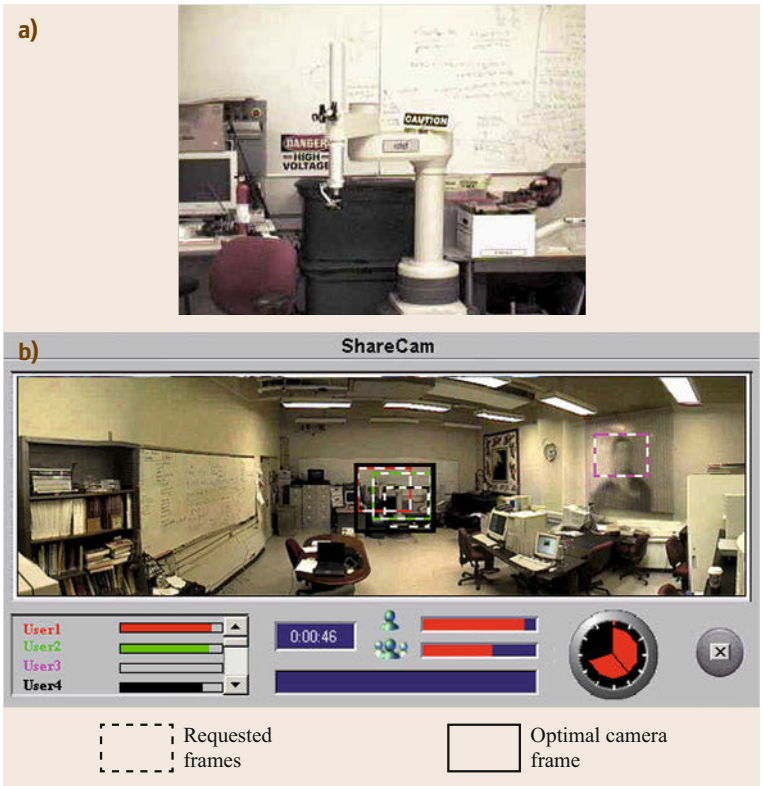


Fig.44.12a,b Frame selection interface (after [44.86]). The user interface includes two image windows. The lower window **(b)** displays a fixed panoramic image based on the camera's full workspace (reachable field of view). Each user requests a camera frame by positioning a *dashed rectangle* in **(b)**. Based on these requests, the algorithm computes an optimal camera frame (shown with a *solid rectangle*), moves the camera accordingly, and displays the resulting live streaming video image in the upper window **(a)**

human inputs can also be replaced by autonomous agents and other sensory inputs. These developments have been applied to a recent project, the Collaborative Observatory for Nature Environments (CONE) project [44.91], which aims to design a networked robotic camera system to collect data from the wilderness for natural scientists.

One important issue in collaborative control is the disconnection between individual commands and the robot action, which may lead to the loss of situation awareness, less participation, and eventual system failure. Inspired by engaging power in scoring systems in computer games, *Goldberg* et al. [44.92] design scoring mechanism for the collaborative control architecture by evaluating the individual leadership level. The early results show great improvement in the group performance. Furthermore, the recent development of social media, such as Blog and Twitter, can also be employed in the collaborative control to facilitate user interaction in real time, which can make the system more engaging and effective. The resulting new architecture can be viewed as a crowd sourcing [44.42, 93] type approach to networked robots that combines human recognition and decision making capabilities to robot execution at a different scale and depth than a regular teleoperation system.

44.5 Cloud Robotics

As noted earlier, the term *Cloud Robotics* is increasingly common based on advances in what is now called *Cloud Computing*. Cloud Robotics extends what were previously called *Online Robots* [44.1] and *Networked Robots* [44.55, 56]. Cloud computing provides robots with vast resources in computation, memory, programming.

Here we review five ways that cloud robotics and automation can potentially improve robots and automation performance:

1. Providing access to global libraries of images, maps, and object data, eventually annotated with geometry and mechanical properties.

2. Massively parallel computation on demand for demanding tasks like optimal motion planning and sample-based statistical modeling
3. Robot sharing of outcomes, trajectories, and dynamic control policies.
4. Human sharing of *open-source* code, data, and designs for programming, experimentation, and hardware construction.
5. On-demand human guidance (*call centers*) for exception handling and error recovery. Updated information and links are available at [44.94].

44.5.1 Big Data

The term *Big Data* describes data sets that are beyond the capabilities of standard relational database systems, which describes the growing library of images, maps, and many other forms of data relevant to robotics and automation on the Internet. One example is grasping, where online datasets can be consulted to determine appropriate grasps. The Columbia Grasp dataset [44.95] and the MIT KIT object dataset [44.96] are available online and have been widely used to evaluate grasping algorithms [44.97–100].

Related work explores how computer vision can be used with Cloud resources to incrementally learn grasp strategies [44.102, 103] by matching sensor data against 3-D computer-aided drafting (CAD) models in an online database. Examples of sensor data include 2-D image features [44.104], 3-D features [44.105], and 3-D point clouds [44.106]. Google Goggles [44.107], a free network-based image recognition service for mobile devices, has been incorporated into a system for robot grasping [44.101] as illustrated in Fig. 44.13.

Dalibard et al. attach *manuals* of manipulation tasks to objects [44.108]. The RoboEarch project stores data related to objects maps, and tasks, for applications ranging from object recognition to mobile navigation to grasping and manipulation (Fig. 44.14) [44.64].

As noted below, online datasets are effectively used to facilitate learning in computer vision. By leveraging Google's 3-D warehouse, [44.109] reduced the need for manually labeled training data. Using community photo collections, [44.110] created an augmented reality application with processing in the cloud.

44.5.2 Cloud Computing

As of 2012, Cloud Computing services like Amazon's EC2 elastic computing engine provide massively parallel computation on demand [44.111]. Examples include Amazon Web Services [44.112] Elastic Compute Cloud, known as EC2 [44.113], Google Compute Engine [44.114], Microsoft Azure [44.115]. These provide a large pool of computing resources that can be rented by the public for short-term computing tasks. These services were originally used primarily by web application developers, but have increasingly been used in scientific and technical high-performance computing (HPC) applications [44.116–119].

Cloud computing is challenging when there are real-time constraints [44.120]; this is an active area of research. However, there are many robotics applications that are not time sensitive such as decluttering a room or precomputing grasp strategies.

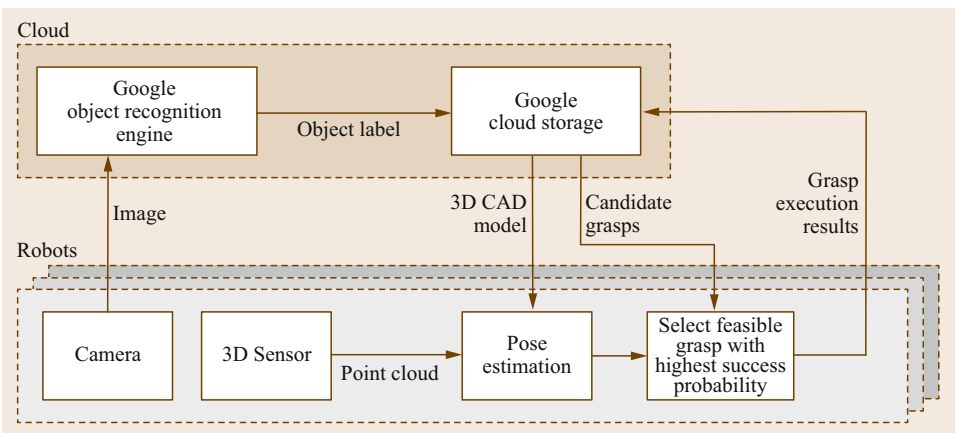


Fig. 44.13 System architecture for cloud-based object recognition for grasping. The robot captures an image of an object and sends via the network to the Google object recognition server. The server processes the image and returns data for a set of candidate objects, each with precomputed grasping options. The robot compares the returned CAD models with the detected point cloud to refine identification and to perform pose estimation, and selects an appropriate grasp. After the grasp is executed, data on the outcome is used to update models in the cloud for future reference (after [44.101])

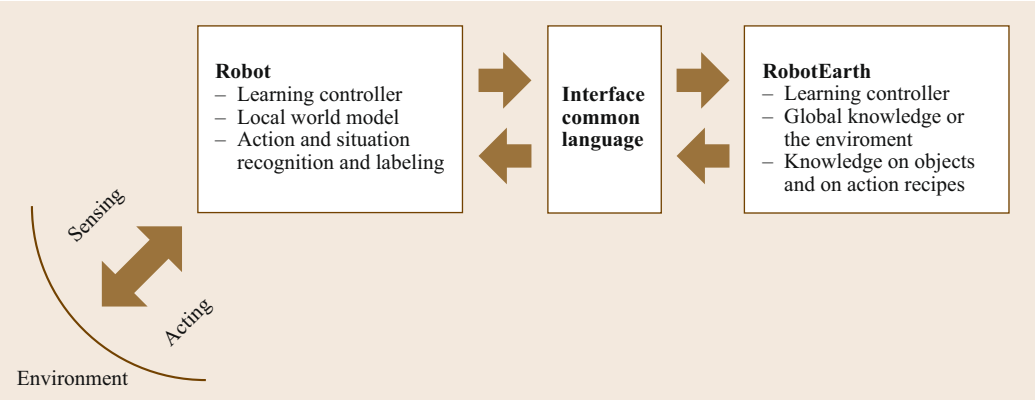


Fig. 44.14 RoboEarth architecture (after [44.64])

There are many sources of uncertainty in robotics and automation [44.121]. Cloud computing allows massive sampling over error distributions and Monte Carlo sampling is *embarrassingly parallel*; recent research in fields as varied as medicine [44.122] and particle physics [44.123] have taken advantage of the cloud. Real-time video and image analysis can be performed in the Cloud [44.109, 124, 125]. Image processing in the cloud has been used for assistive technology for the visually impaired [44.126] and for senior citizens [44.127]. Cloud computing is ideal for sample-based statistical motion planning under uncertainty, where it can be used to explore many possible perturbations in object and environment pose, shape, and robot response to sensors and commands [44.128]. Cloud-based sampling is also being investigated for grasping objects with shape uncertainty [44.129, 130] (Fig. 44.15). A grasp planning algorithm accepts as input a nominal polygonal outline with Gaussian uncertainty around each vertex and the center of mass to compute a grasp quality metric based on a lower bound on the probability of achieving force closure.

44.5.3 Collective Robot Learning

The Cloud allows robots and automation systems to *share* data from physical trials in a variety of environments, for example, initial and desired conditions, associated control policies and trajectories, and importantly: data on performance and outcomes. Such data are a rich source for robot learning.

One example is for path planning, where previously generated paths are adapted to similar environments [44.131] and grasp stability of finger contacts can be learned from previous grasps on an object [44.98].

The MyRobots project [44.132] from RobotShop proposes a *social network* for robots [44.133]:

> *In the same way humans benefit from socializing, collaborating and sharing, robots can benefit from those interactions too by sharing their sensor information giving insight on their perspective of their current state.*

44.5.4 Open Source and Open Access

The Cloud facilitates sharing by humans of designs for hardware, data, and code. The success of open-source software [44.134–136] is now widely accepted in the robotics and automation community. A primary example is ROS, the Robot Operating System, which provides libraries and tools to help software developers create robot applications [44.137, 138]. ROS has also been ported to Android devices [44.139]. ROS has be-

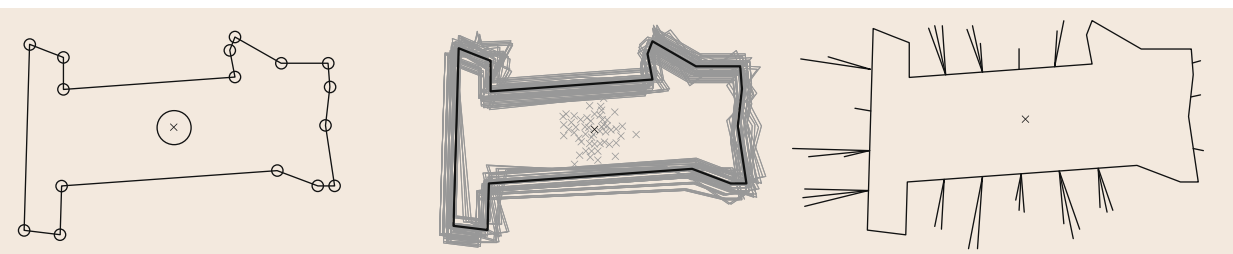


Fig. 44.15 A cloud-based approach to geometric shape uncertainty for grasping (after [44.129, 130])

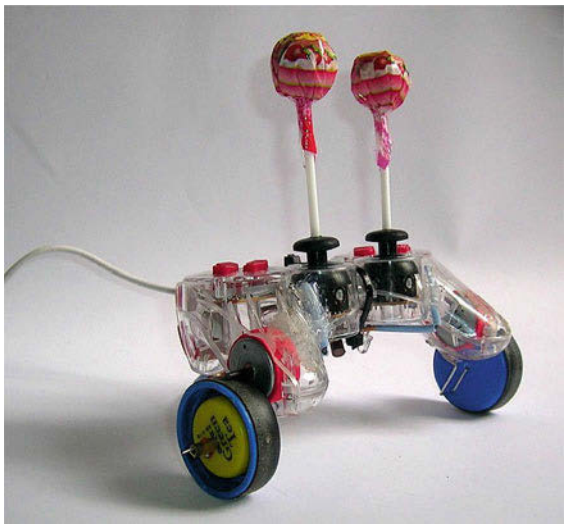


Fig. 44.16 Suckerbot, designed by Tom Tilley of Thailand, a winner of the $10 Robot Design Challenge (after [44.140])

come a standard akin to Linux and is now used by almost all robot developers in research and many in industry.

Additionally, many simulation libraries for robotics are now open-source, which allows students and researchers to rapidly set up and adapt new systems and share the resulting software. Open-source simulation libraries include Bullet [44.141], a physics simulator originally used for video games, OpenRAVE [44.142] and Gazebo [44.143], simulation environments geared specifically toward robotics, OOPSMP, a motion-planning library [44.144], and GraspIt!, a grasping simulator [44.145].

Another exciting trend is in open-source hardware, where CAD models and the technical details of construction of devices are made freely available [44.147, 148]. The Arduino project [44.149] is a widely used open-source microcontroller platform, and has been used in many robotics projects. The Raven [44.150] is an open-source laparoscopic surgery robot developed as a research platform an order of magnitude less expensive than commercial surgical robots [44.151].

The Cloud can also be used to facilitate open challenges and design competitions. For example, the African Robotics Network with support from IEEE Robotics and Automation Society hosted the *$10 Robot* Design Challenge in the summer of 2012. This open competition attracted 28 designs from around the world including a winning entry from Thailand (Fig. 44.16) that modified a surplus Sony game controller, adapting its embedded vibration motors to drive wheels and adding lollipops to the thumb switches as inertial counterweights for contact sensing, which can be built from surplus parts for US $8.96 [44.140].

44.5.5 Crowdsourcing and Call Centers

In contrast to automated telephone reservation and technical support systems, consider a future scenario where errors and exceptions are detected by robots and automation systems, which then access human guidance on-demand at remote call centers. Human skill, experience, and intution is being tapped to solve a number of problems such as image labeling for computer vision [44.54, 62, 102, 152]. Amazon's Mechanical Turk is pioneering on-demand *crowdsourcing* that can draw on *human computation* or *social computing systems*. Research projects are exploring how

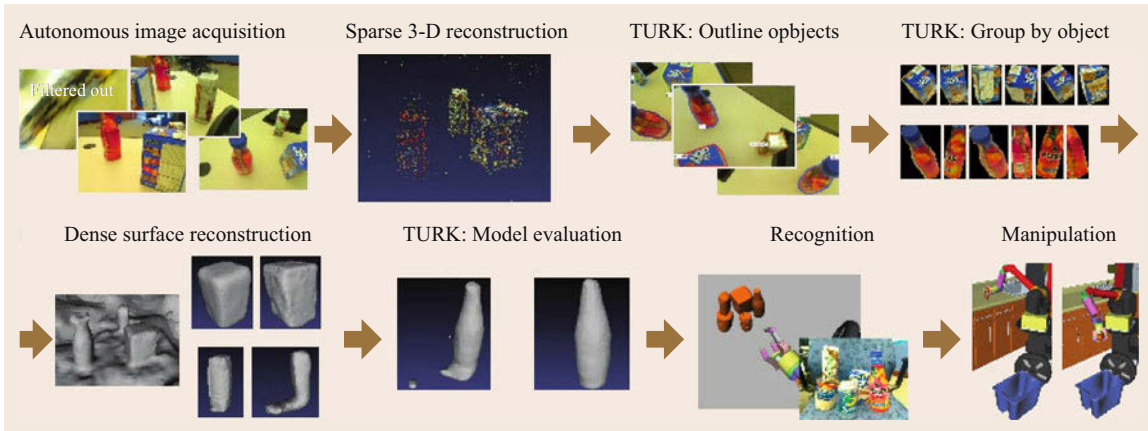


Fig. 44.17 A cloud robot system that incorporates Amazon's Mechanical Turk to *crowdsource* object identification to facilitate robot grasping (after [44.146])

this can be used for path planning [44.153], to determine depth layers, image normals, and symmetry from images [44.154], and to refine image segmentation [44.155]. Researchers are working to understand pricing models [44.156] and apply crowdsourcing to grasping [44.146] (Fig. 44.17).

44.6 Conclusion and Future Directions

As this technology matures, networked robots will gradually go beyond university laboratories and find application in the real world.

As mentioned earlier in Sects. 44.2.2 and 44.5, the new efforts in cloud robotics lead by Google and RoboEarth naturally bridge research and applications. The open source nature and ready-to-use APIs can quickly spread and deploy research results. Japan's Advanced Telecommunications Research Institute International (ATR) Intelligent Robotics and Communication Laboratory has also announced its networked robot project led by Norihiro Hagita (ATR). Its mission is to develop network-based intelligent robots for applications such as service, medical, and safety. Hideyuki Tokuda (Keio University) chaired the Networked Robot Forum in Spring 2005, which promotes research and development (R&D) and standardization on network robots through activities to support awareness campaigns and verification experiments in collaboration among wide-ranging parties, which includes over 100 industry and academic members. Korea's Ministry of Information and Communication has also announced the Ubiquitous Robotic Companion (URC) project to develop network-based intelligent robots.

Networked robots have allowed tens of thousands of nonspecialists around the world to interact with robots. The design of networked robots presents a number of engineering challenges to build reliable systems that can be operated by nonspecialists 24 h a day, 7 days a week, and remain online for years. Many new research challenges remain:

- *New interfaces*: As portable devices such as cellphones and tablet computers increase in computation power, networked robotics should be able to adopt them as new interfaces. As computers becomes increasingly powerful, they become capable of visualizing more sophisticated sensor inputs. Designers of new interfaces should also keep track of new developments in hardware such as haptic interfaces and voice recognition systems. New software standards such as flash, extensible markup language (XML), extensible hyper text markup language (XHTML), virtual reality modeling language (VRML), and wireless markup language (WML) will also change the way we design interface.
  New interface technology arises as human computer interaction technology, mobile computing, and computer graphics areas progress. Recent progresses on brain–machine interaction explore the possibility of using brain wave, such as electroencephalography (EEG) signals, to control robot movements for ground robots [44.157] and unmanned aerial vehicles (UAV)s [44.158]. Gesture [44.159] and multitouch [44.160] are also used to generate control commands. Unlike the traditional mouse and keyboard interfaces, the new interfaces facilitate more natural interaction but suffers from precision issues, because these methods have large noise and require more research efforts in improving robustness and accuracy.
- *New algorithms*: Algorithms determine performance. Scalable algorithms that are capable of handing large amounts of data such as video/sensor network inputs and utilize fast-evolving hardware capability such as distributed and parallel computation will become increasingly important in the networked robotics, especially in cloud robotics.
- *New protocols*: Although we have listed some pioneering work in changing the network environment to improve teleoperation, there are still a large number of open problems such as new protocols, appropriate bandwidth allocation [44.161], QoS [44.162], security, routing mechanisms [44.29, 163], and many more. Network communication is a very fast-evolving field. The incorporation/modification of network communication ideas into networked telerobotic system design will continue to be an active research area. The common object request broker architecture (CORBA) or real-time CORBA [44.20, 21, 39, 164, 165] have great potential for networked robots.
- *New performance metrics*: As more and more robots enter service, it is important to develop metrics to quantify the performance of the robot–human team. As we are more familiar with metrics developed to assess robot performance or task performance [44.162], recent progresses on using the robot to assess human performance [44.166,

167] shed light on new metrics. Standardizing these metrics will also be an important direction.

- *Video for robotics*: Another interesting observation is that all of existing video compression and transmission standards try to rebuild a true and complete representation of camera field of view. However, it might not be necessary or infeasible due to bandwidth limit for a networked robot [44.168]. Sometimes, a high level abstraction is sufficient. For example, when a mobile robot is avoiding an moving obstacle, all the robot needs to know is the speed and bounding box of the moving object instead of knowledge that whether this object is human or other robots. We might want to control the level of details in video perception and transmission. This actually imposes an interesting problem: we need a new streaming standard that serves for networked robots.
- *Applications*: Recent successful applications include environment monitoring [44.43, 169], manufacturing [44.170, 171], and infrastructure inspection and maintenance [44.172, 173]. The fast development of networked robot systems is worldwide. Many new applications are emerging in areas such as security, inspection, education, and entertainment. Application requirements such as reliability, security, and modularity will continue to pose new challenges for system design.

Video-References

VIDEO 81 A heterogeneous multiple-operator-multiple-robot system
available from http://handbookofrobotics.org/view-chapter/44/videodetails/81

VIDEO 82 Teleoperation of a mini-excavator
available from http://handbookofrobotics.org/view-chapter/44/videodetails/82

VIDEO 83 Tele-Actor
available from http://handbookofrobotics.org/view-chapter/44/videodetails/83

VIDEO 84 A multi-operator-multi-robot teleoperation system
available from http://handbookofrobotics.org/view-chapter/44/videodetails/84

References

44.1 K. Goldberg, R. Siegwart (Eds.): *Beyond Webcams: An Introduction to Online Robots* (MIT Press, Cambridge 2002)

44.2 IEEE Technical Committee on Networked Robots: http://tab.ieee-ras.org/

44.3 N. Tesla: Method of and apparatus for controlling mechanism of moving vessels or vehicles, US Patent 613809 A (1898)

44.4 R. Goertz, R. Thompson: Electronically controlled manipulator, Nucleonics **12**(11), 46–47 (1954)

44.5 R.D. Ballard: A last long look at titanic, Nat. Geogr. **170**(6), 698–727 (1986)

44.6 A.K. Bejczy: Sensors, controls, and man-machine interface for advanced teleoperation, Science **208**(4450), 1327–1335 (1980)

44.7 R.S. Mosher: Industrial manipulators, Sci. Am. **211**(4), 88–96 (1964)

44.8 R. Tomovic: On man-machine control, Automatica **5**, 401–404 (1969)

44.9 A. Bejczy, G. Bekey, R. Taylor, S. Rovetta: A research methodology for tele-surgery with time delays, 1st Int. Symp. Med. Robotics Comput. Assist. Surg. (MRCAS) (1994)

44.10 M. Gertz, D. Stewart, P. Khosla: A human-machine interface for distributed virtual laboratories, IEEE Robotics Autom. Mag. **1**, 5–13 (1994)

44.11 T. Sato, J. Ichikawa, M. Mitsuishi, Y. Hatamura: A new micro-teleoperation system employing a hand-held force feedback pencil, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994)

44.12 T.B. Sheridan: *Telerobotics, Automation, and Human Supervisory Control* (MIT Press, Cambridge 1992)

44.13 FirstWebcam: http://www.cl.cam.ac.uk/coffee/qsf/timeline.html. (1993)

44.14 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Robot teleoperation via WWW, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995)

44.15 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Beyond the web: Manipulating the physical world via the WWW, Comput. Netw. ISDN Syst. J. **28**(1), 209–219 (1995)

44.16 B. Dalton, K. Taylor: A framework for internet robotics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998)

44.17 K. Taylor, J. Trevelyan: The telelabs project, http://telerobot.mech.uwa.edu.au (1994)

44.18 H. Hu, L. Yu, P.W. Tsui, Q. Zhou: Internet-based robotic systems for teleoperation, Assem. Automat. **21**(2), 143–151 (2001)

44.19 R. Safaric, M. Debevc, R. Parkin, S. Uran: Telerobotics experiments via internet, IEEE Trans. Ind. Electron. **48**(2), 424–431 (2001)

44.20 S. Jia, K. Takase: A corba-based internet robotic system, Adv. Robotics **15**(6), 663–673 (2001)

44.21 S. Jia, Y. Hada, G. Ye, K. Takase: Distributed telecare robotic systems using corba as a communication architecture, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.22 J. Kim, B. Choi, S. Park, K. Kim, S. Ko: Remote control system using real-time MPEG-4 streaming technology for mobile robot, IEEE Int. Conf. Consum. Electron. (2002)

44.23 T. Mirfakhrai, S. Payandeh: A delay prediction approach for teleoperation over the internet, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.24 K. Han, Y. Kim, J. Kim, S. Hsia: Internet control of personal robot between KAIST and UC Davis, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.25 L. Ngai, W.S. Newman, V. Liberatore: An experiment in internet-based, human-assisted robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.26 R.C. Luo, T.M. Chen: Development of a multibehavior-based mobile robot for remote supervisory control through the internet, IEEE/ASME Trans. Mechatron. **5**(4), 376–385 (2000)

44.27 D. Aarno, S. Ekvall, D. Kragi: Adaptive virtual fixtures for machine-assisted teleoperation tasks, IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1151–1156

44.28 I. Belousov, S. Chebukov, V. Sazonov: Web-based teleoperation of the robot interacting with fast moving objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 685–690

44.29 Z. Cen, A. Goradia, M. Mutka, N. Xi, W. Fung, Y. Liu: Improving the operation efficiency of supermedia enhanced internet based teleoperation via an overlay network, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 691–696

44.30 N.P. Jouppi, S. Thomas: Telepresence systems with automatic preservation of user head height, local rotation, and remote translation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 62–68

44.31 B. Ricks, C.W. Nielsen, M.A. Goodrich: Ecological displays for robot interaction: A new perspective, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (2004) pp. 2855–2860

44.32 D. Ryu, S. Kang, M. Kim, J. Song: Multi-modal user interface for teleoperation of ROBHAZ-DT2 field robot system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 1 (2004) pp. 168–173

44.33 J. Su, Z. Luo: Incremental motion compression for telepresent walking subject to spatial constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 69–74

44.34 I. Toshima, S. Aoki: Effect of driving delay with an acoustical tele-presence robot, telehead, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 56–61

44.35 N. Chong, T. Kotoku, K. Ohba, K. Komoriya, N. Matsuhira, K. Tanie: Remote coordinated controls in multiple telerobot cooperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (2000) pp. 3138–3343

44.36 P. Cheng, V. Kumar: An almost communicationless approach to task allocation for multiple unmanned aerial vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1384–1389

44.37 X. Ding, M. Powers, M. Egerstedt, S. Young, T. Balch: Executive decision support, IEEE Robotics Autom. Mag. **16**(2), 73–81 (2009)

44.38 J. Liu, L. Sun, T. Chen, X. Huang, C. Zhao: Competitive multi-robot teleoperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

44.39 Z. Zhang, Q. Cao, L. Zhang, C. Lo: A CORBA-based cooperative mobile robot system, Ind. Robot Int. J. **36**(1), 36–44 (2009)

44.40 Y. Xu, D. Song: Systems and algorithms for autonomous and scalable crowd surveillance using robotic PTZ cameras assisted by a wide-angle camera, Auton. Robots **29**(1), 53–66 (2010)

44.41 D. Sanders, J. Graham-Jones, A. Gegov: Improving ability of tele-operators to complete progressively more difficult mobile robot paths using simple expert systems and ultrasonic sensors, Ind. Robot Int. J. **37**(5), 431–440 (2010)

44.42 S. Faridani, B. Lee, S. Glasscock, J. Rappole, D. Song, K. Goldberg: A networked telerobotic observatory for collaborative remote observation of avian activity and range change, IFAC Workshop Netw. Robots (2009)

44.43 R. Bogue: Robots for monitoring the environment, Ind. Robot Int. J. **38**(6), 560–566 (2011)

44.44 R. Murphy, J. Burke: From remote tool to shared roles, IEEE Robotics Autom. Mag. **15**(4), 39–49 (2008)

44.45 E. Paulos, J. Canny, F. Barrientos: Prop: Personal roving presence, SIGGRAPH Vis. Proc. (1997) p. 99

44.46 L. Takayama, E. Marder-Eppstein, H. Harris, J. Beer: Assisted driving of a mobile remote presence system: System design and controlled user evaluation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1883–1889

44.47 D. Lazewatsky, W. Smart: An inexpensive robot platform for teleoperation and experimentation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1211–1216

44.48 K. Taylor, J.P. Trevelyan: Australia's telerobot on the web, 26th Symp. Ind. Robotics (1995) pp. 39–44

44.49 A. Khamis, D.M. Rivero, F. Rodriguez, M. Salichs: Pattern-based architecture for building mobile robotics remote laboratories, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3284–3289

44.50 C. Cosma, M. Confente, D. Botturi, P. Fiorini: Laboratory tools for robotics and automation education, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3303–3308

44.51 K.W. Dorman, J.L. Pullen, W.O. Keksz, P.H. Eismann, K.A. Kowalski, J.P. Karlen: The servicing aid tool: A teleoperated robotics system for space applications, 7th Annu. Workshop Space Operat. Appl. Res. (SOAR), Vol. 1 (1994)

Avidbots Ex1016
Page 1156 of 2154

44.52 C. Pollak, H. Hutter: A webcam as recording device for light microscopes, J. Comput.-Assist. Microsc. **10**(4), 179–183 (1998)

44.53 K. Goldberg, D. Song, A. Levandowski: Collaborative teleoperation using networked spatial dynamic voting, Proceedings IEEE **91**(3), 430–439 (2003)

44.54 J.J. Kuffner: Cloud-Enabled Robots, IEEE-RAS Int. Conf. Humanoid Robots (2010)

44.55 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Desktop teleoperation via the World Wide Web, Proc. IEEE Int. Conf. Robotics Autom., Vol. 1 (1995) pp. 654–659

44.56 G. McKee: What is networked robotics?, Inf. Control Autom. Robotics **15**, 35–45 (2008)

44.57 E. Guizzo: Cloud robotics: Connected to the cloud, robots get smarter, IEEE Spectrum http://spectrum.ieee.org/automaton/robotics/robotics-software/cloud-robotics (2011)

44.58 M. Tenorth, A.C. Perzylo, R. Lafrenz, M. Beetz: The RoboEarth language: Representing and exchanging knowledge about actions, objects, and environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1284–1289

44.59 R. Arumugam, V.R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F.F. Kong, A.S. Kumar, K.D. Meng, G.W. Kit: DAvinCi: A cloud computing framework for service robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 3084–3089

44.60 Z. Du, W. Yang, Y. Chen, X. Sun, X. Wang, C. Xu: Design of a robot cloud center, Int. Symp. Auton. Decentralized Syst. (2011) pp. 269–275

44.61 G. Hu, W.P. Tay, Y. Wen: Cloud robotics: Architecture, challenges and applications, IEEE Network **26**(3), 21–28 (2012)

44.62 K. Kamei, S. Nishio, N. Hagita, M. Sato: Cloud networked robotics, IEEE Network **26**(3), 28–34 (2012)

44.63 D. Hunziker, M. Gajamohan, M. Waibel, R. D'Andrea: Rapyuta: The RoboEarth cloud engine, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

44.64 M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Gálvez-López, K. Häussermann, R. Janssen, J.M.M. Montiel, A. Perzylo, B. le Schieß, M. Tenorth, O. Zweigle, R. De Molengraft: RoboEarth, IEEE Robotics Autom. Mag. **18**(2), 69–82 (2011)

44.65 RoboEarth: What is RoboEarth?, http://www.roboearth.org/what-is-roboearth

44.66 L. Atzori, A. Iera, G. Morabito: The internet of things: A survey, Comput. Netw. **54**(15), 2787–2805 (2010)

44.67 J. Walrand, P. Varaiya: *High-Performance Communication Networks*, 2nd edn. (Morgan Kaufmann Press, San Francisco 2000)

44.68 M.A. Peshkin, A.C. Sanderson: Minimization of energy in quasi-static manipulation, IEEE Trans. Robotics Autom. **5**(1), 53–60 (1989)

44.69 M.T. Mason: On the scope of quasi-static pushing, 3rd Int. Symp. Robotics Res. (1986)

44.70 E. Ladd, J. O'Donnell: *Using Html 4, Xml, and Java 1.2* (QUE Press, Indianapolis 1998)

44.71 H. Friz: Design of an Augmented Reality User Interface for an Internet based Telerobot using Multiple Monoscopic Views, Ph.D. Thesis (Technical Univ. Clausthal, Clausthal-Zellerfeld 2000)

44.72 T. Fong, C. Thorpe: Vehicle teleoperation interfaces, Auton. Robots **11**, 9–18 (2001)

44.73 A. Birk, N. Vaskevicius, K. Pathak, S. Schwertfeger, J. Poppinga, H. Buelow: 3-D perception and modeling, IEEE Robotics Autom. Mag. **16**(4), 53–60 (2009)

44.74 A. Kellyo, N. Chan, H. Herman, D. Huber, R. Meyers, P. Rander, R. Warner, J. Ziglar, E. Capstick: Real-time photorealistic virtualized reality interface for remote mobile robot control, Int. J. Robotics Res. **30**(3), 384–404 (2011)

44.75 L. Conway, R.A. Volz, M.W. Walker: Teleautonomous systems: Projecting and coordinating intelligent action at a distance, IEEE Trans. Robotics Autom. **6**(20), 146–158 (1990)

44.76 J. Larsson, M. Broxvall, A. Saffiotti: An evaluation of local autonomy applied to teleoperated vehicles in underground mines, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 1745–1752

44.77 L.B. Rosenberg: Virtual fixtures: Perceptual tools for telerobotic manipulation, IEEE Virtual Real. Annu. Int. Symp. (VRAIS) (1993) pp. 76–82

44.78 P. Marayong, M. Li, A. Okamura, G. Hager: Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2003) pp. 1954–1959

44.79 A. Bettini, P. Marayong, S. Lang, A. Okamura, G. Hager: Vision-assisted control for manipulation using virtual fixtures, IEEE Trans. Robotics **20**(6), 953–966 (2004)

44.80 R. Azuma: A survey of augmented reality, Presence **6**(4), 355–385 (1997)

44.81 K. Goldberg, B. Chen, R. Solomon, S. Bui, B. Farzin, J. Heitler, D. Poon, G. Smith: Collaborative teleoperation via the internet, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2000) pp. 2019–2024

44.82 D. Song: Systems and Algorithms for Collaborative Teleoperation, Ph.D. Thesis (Univ. California, Berkeley 2004)

44.83 D. Song: *Sharing a Vision: Systems and Algorithms for Collaboratively-Teleoperated Robotic Cameras* (Springer, Berlin, Heidelberg 2009)

44.84 K. Goldberg, B. Chen: Collaborative teleoperation via the internet, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001)

44.85 K. Goldberg, D. Song: Tele-Actor http://www.tele-actor.net, Univ. of California, Berkeley

44.86 D. Song, A. Pashkevich, K. Goldberg: Sharecam part II: Approximate and distributed algorithms for a collaboratively controlled robotic webcam, Proc. IEEE/RSJ Int. Conf. Intell. Robots (IROS), Vol. 2 (2003) pp. 1087–1093

44.87 D. Song, K. Goldberg: Sharecam part I: Interface, system architecture, and implementation of a collaboratively controlled robotic webcam, Proc.

IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (2003) pp. 1080–1086
44.88 D. Song, K. Goldberg: Approximate algorithms for a collaboratively controlled robotic camera, IEEE Trans. Robotics **23**(5), 1061–1070 (2007)
44.89 D. Song, A.F. van der Stappen, K. Goldberg: Exact algorithms for single frame selection on multi-axis satellites, IEEE Trans. Autom. Sci. Eng. **3**(1), 16–28 (2006)
44.90 Y. Xu, D. Song, J. Yi: An approximation algorithm for the least overlapping p-frame problem with non-partial coverage for networked robotic cameras, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008)
44.91 D. Song, N. Qin, K. Goldberg: Systems, control models, and codec for collaborative observation of remote environments with an autonomous networked robotic camera, Auton. Robots **24**(4), 435–449 (2008)
44.92 K. Goldberg, D. Song, I.Y. Song, J. McGonigal, W. Zheng, D. Plautz: Unsupervised scoring for scalable internet-based collaborative teleoperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)
44.93 J. Rappole, S. Glasscock, K. Goldberg, D. Song, S. Faridani: Range change among new world tropical and subtropical birds, Bonn. Zool. Monogr. **57**, 151–167 (2011)
44.94 http://goldberg.berkeley.edu/cloud-robotics/, UC Berkeley
44.95 C. Goldfeder, M. Ciocarlie, P.K. Allen: The Columbia grasp database, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 1710–1716
44.96 A. Kasper, Z. Xue, R. Dillmann: The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics, Int. J. Robotics Res. **31**(8), 927–934 (2012)
44.97 H. Dang, J. Weisz, P.K. Allen: Blind grasping: Stable robotic grasping using tactile feedback and hand kinematics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5917–5922
44.98 H. Dang, P.K. Allen: Learning grasp stability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 2392–2397
44.99 J. Weisz, P.K. Allen: Pose error robust grasping from contact wrench space metrics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 557–562
44.100 M. Popovic, G. Kootstra, J.A. Jorgensen, D. Kragic, N. Kruger: Grasping unknown objects using an Early Cognitive Vision system for general scene understanding, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 987–994
44.101 B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, K. Goldberg: Cloud-based robot grasping with the Google object recognition engine, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
44.102 M. Ciocarlie, C. Pantofaru, K. Hsiao, G. Bradski, P. Brook, E. Dreyfuss: A side of data with my robot, IEEE Robotics Autom. Mag. **18**(2), 44–57 (2011)
44.103 M.A. Moussa, M.S. Kamel: An experimental approach to robotic grasping using a connectionist architecture and generic grasping functions, IEEE Trans. Syst. Man Cybern. C **28**(2), 239–253 (1998)
44.104 K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic: Grasping known objects with humanoid robots: A box-based approach, Int. Conf. Adv. Robotics (2009)
44.105 C. Goldfeder, P.K. Allen: Data-driven grasping, Auton. Robots **31**(1), 1–20 (2011)
44.106 M. Ciocarlie, K. Hsiao, E.G. Jones, S. Chitta, R.B. Rusu, I.A. Sucan: Towards reliable grasping and manipulation in household environments, Intl. Symp. Exp. Robotics (2010) pp. 1–12
44.107 Google Goggles, http://www.google.com/mobile/goggles/
44.108 S. Dalibard, A. Nakhaei, F. Lamiraux, J.-P. Laumond: Manipulation of documented objects by a walking humanoid robot, IEEE-RAS Int. Conf. Humanoid Robots (2010) pp. 518–523
44.109 K. Lai, D. Fox: Object recognition in 3-D point clouds using web data and domain adaptation, Int. J. Robotics Res. **29**(8), 1019–1037 (2010)
44.110 S. Gammeter, A. Gassmann, L. Bossard, T. Quack, L. Van Gool: Server-side object recognition and client-side object tracking for mobile augmented reality, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (2010) pp. 1–8
44.111 M. Armbrust, I. Stoica, M. Zaharia, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin: A view of cloud computing, Communication ACM **53**(4), 50 (2010)
44.112 Amazon Web Services, http://aws.amazon.com
44.113 Amazon Elastic Cloud (EC2), http://aws.amazon.com/ec2/
44.114 Google Compute Engine, https://cloud.google.com/products/compute-engine
44.115 Microsoft Azure, http://www.windowsazure.com
44.116 G. Juve, E. Deelman, G.B. Berriman, B.P. Berman, P. Maechling: An evaluation of the cost and performance of scientific workflows on Amazon EC2, J. Grid Comput. **10**(1), 5–21 (2012)
44.117 P. Mehrotra, J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, S. Saini, R. Biswas: Performance evaluation of Amazon EC2 for NASA HPC applications, Proc. 3rd Workshop Sci. Cloud Comput. Date (ScienceCloud) (2012)
44.118 R. Tudoran, A. Costan, G. Antoniu, L. Bougé: A performance evaluation of Azure and Nimbus clouds for scientific applications, Proc. 2nd Int. Workshop Cloud Comput. Platf. (CloudCP) (2012) pp. 1–6
44.119 TOP500 List, http://www.top500.org/list/2012/06/100 (June 2012)
44.120 N.K. Jangid: Real time cloud computing, Proc. 1st Natl. Conf. Data Manag. Secur., Jaipur (2011)
44.121 J. Glover, D. Rus, N. Roy: Probabilistic models of object geometry for grasp planning. In: *Robotics: Science and Systems IV*, ed. by O. Brock, J. Trinkle, F. Ramos (MIT Press, Cambridge 2008)
44.122 H. Wang, Y. Ma, G. Pratx, L. Xing: Toward real-time Monte Carlo simulation using a commercial cloud computing infrastructure, Phys. Med. Biol. **56**(17), 175–181 (2011)

44.123 M. Sevior, T. Fifield, N. Katayama: Belle monte-carlo production on the Amazon EC2 cloud, J. Phys. **219**(1), 012003 (2010)

44.124 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, IEEE Comput. Soc. Conf. Comp. Vis. Pattern Recognit., Vol. 2 (2006) pp. 2161–2168

44.125 J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman: Object retrieval with large vocabularies and fast spatial matching, IEEE Conf. Comput. Vis. Pattern Recognit. (2007) pp. 1–8

44.126 B. Bhargava, P. Angin, L. Duan: A mobile-cloud pedestrian crossing guide for the blind, Int. Conf. Adv. Comput. Commun. (2011)

44.127 J.J.S. García: Using cloud computing as a HPC platform for embedded systems, http://www.atc.us.es/descargas/tfmHPCCloud.pdf (2011)

44.128 J. van den Berg, P. Abbeel, K. Goldberg: LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information, Int. J. Robotics Res. **30**(7), 895–913 (2011)

44.129 B. Kehoe, D. Berenson, K. Goldberg: Estimating part tolerance bounds based on adaptive cloud-based grasp planning with slip, Proc. IEEE Int. Conf. Automat. Sci. Eng. (2012)

44.130 B. Kehoe, D. Berenson, K. Goldberg: Toward cloud-based grasping with uncertainty in shape: Estimating lower bounds on achieving force closure with zero-slip push grasps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 576–583

44.131 D. Berenson, P. Abbeel, K. Goldberg: A robot path planning framework that learns from experience, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3671–3678

44.132 MyRobots.com, http://myrobots.com

44.133 What is MyRobots?, http://myrobots.com/wiki/About

44.134 L. Dabbish, C. Stuart, J. Tsay, J. Herbsleb: Social coding in GitHub: transparency and collaboration in an open software repository, Proc. ACM Conf. Comp. Support. Coop. Work (2012) pp. 1277–1286

44.135 A. Hars: Working for free? Motivations of participating in open source projects, Proc. 34th Annu. Hawaii Int. Conf. Syst. Sci. (2001)

44.136 D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov: The Eucalyptus open-source cloud-computing system, IEEE/ACM Int. Symp. Clust. Comput. Grid (2009) pp. 124–131

44.137 ROS (Robot Operating System), http://ros.org.

44.138 M. Quigley, B. Gerkey: ROS: An open-source robot operating system, ICRA Workshop Open Source Softw. (2009)

44.139 rosjava, an implementation of ROS in pure Java with Android support, http://cloudrobotics.com

44.140 The African Robotics Network (AFRON): *Ten Dollar Robot* design challenge winners, http://robotics-africa.org/design_challenge.html (2012)

44.141 Bullet Physics Library, http://bulletphysics.org

44.142 OpenRAVE, http://openrave.org/

44.143 Gazebo, http://gazebosim.org

44.144 E. Plaku, K.E. Bekris, L.E. Kavraki: OOPS for motion planning: An online, open-source, programming system, Proc. IEEE Int. Conf. Robotics Autom. (2007) pp. 3711–3716

44.145 A.T. Miller, P.K. Allen: GraspIt! A versatile simulator for robotic grasping, IEEE Robotics Autom. Mag. **11**(4), 110–122 (2004)

44.146 A. Sorokin, D. Berenson, S.S. Srinivasa, M. Hebert: People helping robots helping people: Crowdsourcing for grasping novel objects, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2117–2122

44.147 S. Davidson: Open-source hardware, IEEE Des. Test Comput. **21**(5), 456–456 (2004)

44.148 E. Rubow: Open Source Hardware, Tech. Rep. http://cseweb.ucsd.edu/classes/fa08/cse237a/topicresearch/erubow_tr_report.pdf (2008)

44.149 Arduino: http://www.arduino.cc

44.150 H.H. King, L. Cheng, P. Roan, D. Friedman, S. Nia, J. Ma, D. Glozman, J. Rosen, B. Hannaford: Raven II: Open platform for surgical robotics research, Hamlyn Symp. Med. Robotics (2012)

44.151 An open-source robo-surgeon, The Economist, http://www.economist.com/node/21548489 (2012)

44.152 L. von Ahn: Human computation, Des. Autom. Conf. (2009) pp. 418–419

44.153 J.C. Gamboa Higuera, A. Xu, F. Shkurti, G. Dudek: Socially-driven collective path planning for robot missions, 9th Conf. Comput. Robot Vis. (2012) pp. 417–424

44.154 Y. Gingold, A. Shamir, D. Cohen-Or: Micro perceptual human computation for visual tasks, ACM Trans. Graphics **31**(5), 1–12 (2012)

44.155 M. Johnson-Roberson, J. Bohg, G. Skantze, J. Gustafson, R. Carlson, B. Rasolzadeh, D. Kragic: Enhanced visual scene understanding through human-robot dialog, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3342–3348

44.156 A. Sorokin, D. Forsyth: Utility data annotation with Amazon Mechanical Turk, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (2008) pp. 1–8

44.157 C. Escolano, J. Antelis, J. Minguez: Human brain-teleoperated robot between remote places, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 4430–4437

44.158 A. Akce, M. Johnson, T. Bretl: Remote teleoperation of an unmanned aircraft with a brain-machine interface: Theory and preliminary results, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 5322–5327

44.159 J. Roselln, R. Suárez, C. Rosales, A. Pérez: Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures, Auton. Robots **31**(1), 87–102 (2011)

44.160 K. Onda, F. Arai: Parallel teleoperation of holographic optical tweezers using multi-touch user interface, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1069–1074

44.161 P.X. Liu, M. Meng, S.X. Yang: Data communications for internet robots, Auton. Robots **15**, 213–223 (2003)

44.162 W. Fung, N. Xi, W. Lo, B. Song, Y. Sun, Y. Liu, I.H. Elhajj: Task driven dynamic QOS based bandwidth allcoation for real-time teleoperation via the internet, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003)

44.163 F. Zeiger, N. Kraemer, K. Schilling: Commanding mobile robots via wireless ad-hoc networks – A comparison of four ad-hoc routing protocol implementations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 590–595

44.164 M. Amoretti, S. Bottazzi, M. Reggiani, S. Caselli: Evaluation of data distribution techniques in a corba-based telerobotic system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003)

44.165 S. Bottazzi, S. Caselli, M. Reggiani, M. Amoretti: A software framework based on real time COBRA for telerobotics systems, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002)

44.166 J. Chen, E. Haas, M. Barnes: Human performance issues and user interface design for teleoperated robots, IEEE Trans. Syst. Man Cybern. C **37**(6), 1231–1245 (2007)

44.167 Y. Jia, N. Xi, Y. Wang, X. Li: Online identification of quality of teleoperator (QoT) for performance improvement of telerobotic operations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 451–456

44.168 S. Livatino, G. Muscato, S. Sessa, C. Koffel, C. Arena, A. Pennisi, D. Di Mauro, F. Malkondu: Mobile robotic teleguide based on video images, IEEE Robotics Autom. Mag. **15**(4), 58–67 (2008)

44.169 G. Podnar, J. Dolan, A. Elfes, S. Stancliff, E. Lin, J. Hosier, T. Ames, J. Moisan, T. Moisan, J. Higinbotham, E. Kulczycki: Operation of robotic science boats using the telesupervised adaptive ocean sensor fleet system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1061–1068

44.170 Y. Kwon, S. Rauniar: E-quality for manufacturing (EQM) within the framework of internet-based systems, IEEE Trans. Syst. Man Cybern. C **37**(6), 1365–1372 (2007)

44.171 L. Wang: Wise-shopfloor: An integrated approach for web-based collaborative manufacturing, IEEE Trans. Syst. Man Cybern. C **38**(4), 562–573 (2008)

44.172 P. Debenest, M. Guarnieri, K. Takita, E. Fukushima, S. Hirose, K. Tamura, A. Kimura, H. Kubokawa, N. Iwama, F. Shiga: Expliner – Robot for inspection of transmission lines, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3978–3984

44.173 N. Pouliot, S. Montambault: Linescout technology: From inspection to robotic maintenance on live transmission power lines, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 1034–1040

Avidbots Ex1016
Page 1160 of 2154

Multimedia Contents



Part E Moving in the Environment

Ed. by Raja Chatila

45 World Modeling
Wolfram Burgard, Freiburg, Germany
Martial Hebert, Pittsburgh, USA
Maren Bennewitz, Bonn, Germany

46 Simultaneous Localization and Mapping
Cyrill Stachniss, Bonn, Germany
John J. Leonard, Cambridge, USA
Sebastian Thrun, Mountain View, USA

47 Motion Planning and Obstacle Avoidance
Javier Minguez, Zaragoza, Spain
Florant Lamiraux, Toulouse, France
Jean-Paul Laumond, Toulouse, France

48 Modeling and Control of Legged Robots
Pierre-Brice Wieber, Grenoble, France
Russ Tedrake, Cambridge, USA
Scott Kuindersma, Cambridge, USA

49 Modeling and Control of Wheeled Mobile Robots
Claude Samson, Sophia-Antipolis, France
Pascal Morin, Paris, France
Roland Lenain, Aubiere, France

50 Modeling and Control of Robots on Rough Terrain
Keiji Nagatani, Sendai, Japan
Genya Ishigami, Yokohama, Japan
Yoshito Okada, Sendai, Japan

51 Modeling and Control of Underwater Robots
Gianluca Antonelli, Cassino, Italy
Thor I. Fossen, Trondheim, Norway
Dana R. Yoerger, Woods Hole, USA

52 Modeling and Control of Aerial Robots
Robert Mahony, Canberra, Australia
Randal W. Beard, Provo, USA
Vijay Kumar, Philadelphia, USA

53 Multiple Mobile Robot Systems
Lynne E. Parker, Knoxville, USA
Daniela Rus, Cambridge, USA
Gaurav S. Sukhatme, Los Angeles, USA

Until the mid 1960s, robots were only able to move in a predetermined workspace, the one they could reach from their firmly fixed base. Part E is about their conquest of the whole space. Mobile Robotics started as a research domain in its own right in the late 1960s with the Shakey project at SRI. 2015 marks the 50th anniversary of this seminal project that had a lasting legacy such as A^*. The seminal paper by N.J. Nilsson *A Mobile Automaton: An Application of Artificial Intelligence Techniques* at the International Joint Conference on Artificial Intelligence (IJCAI) 1969, already addressed perception, mapping, motion planning, and the notion of control architecture. Those issues would indeed be at the core of mobile robotics research for the following decades. The 1980s boomed with mobile robot projects, and as soon as it was necessary to cope with the reality of the real physical world, problems appeared that fostered novel research directions, actually moving away from the original concept in which the robot was just an application of artificial intelligence (AI) techniques. This part addresses all the issues that, put together, are necessary to build and control a mobile robot, except for the mechanical design itself.

Navigation is the capacity of moving from one location to another arbitrarily distant one. To move efficiently, a robot needs to use appropriate representations of its environment in order to plan and control its motions according to the presence of obstacles and terrain difficulties, and also to use environment features as landmarks for its localization. This is the topic of Chapter 45 which addresses the different representations used for indoors or outdoors environments, including topological maps and semantic attributes.

However, building environment maps is usually achieved incrementally, as the robot discovers its environment while navigating in it. Hence partial perceptions built from different positions, need to be fused together – taking into account sensing errors and uncertainties (see Chapter 5, Part A) to construct a consistent global map. This requires the robot to know these positions, and, because of motion inaccuracies, the transforms between them are uncertain. This requires to reference the positions to environment features which are only defined in the environment map itself. As a result, localization needs the map and mapping needs localization. Hence, localization and mapping are two interwound problems that must be solved simultaneously. The solution to this problem is the topic of Chapter 46.

Once environment maps are available, or during their construction, the robot has to plan its path as optimally as possible to reach its targets while avoiding obstacles. Chapter 7 in Part A overviews the techniques for solving this central problem, which requires geometrical reasoning in the configuration space (CS) whose construction is complex. To avoid an explicit construction of the CS, probabilistic techniques proved the most efficient way to compute paths – at the price of optimality. However, the kinematic constraints of the robot's locomotion system such as non-holonomy have to be taken into account. We see in Chapter 47 in this part how control problems cannot be separated from geometry. This chapter addresses motion planning from the viewpoint of mobile robotics and introduces the tools from control theory and differential geometry that enable to tackle kinematics constraints. In addition, it provides an overview of local sensor-based methods that are used when the robot encounters unknown or mobile obstacles while moving.

The rest of this part essentially focusses on various means of locomotion used to move in the environment: legged locomotion (Chapter 48), robots with wheels (Chapter 49), with tracks (Chapter 50), and underwater and aerial robots (Chapters 51 and 52 respectively) which face the specific problems of three-dimensional (3-D) motion with environment perturbations. In these two last cases the close links between Robotics and Control are of course central.

Finally, after explaining how one single robot moves in different environments, Part E concludes with Chapter 53 on the interaction and coordination of multiple mobile robot systems.

Multimedia Contents

45. World Modeling

Wolfram Burgard, Martial Hebert, Maren Bennewitz

In this chapter we describe popular ways to represent the environment of a mobile robot. For indoor environments, which are often stored using two-dimensional representations, we discuss occupancy grids, line maps, topological maps, and landmark-based representations. Each of these techniques has its own advantages and disadvantages. Whilst occupancy grid maps allow for quick access and can efficiently be updated, line maps are more compact. Also landmark-based maps can efficiently be updated and maintained, however, they do not readily support navigation tasks such as path planning like topological representations do.

Additionally, we discuss approaches suited for outdoor terrain modeling. In outdoor environments, the flat-surface assumption underling many mapping techniques for indoor environments is no longer valid. A very popular approach in this context are elevation and variants maps, which store the surface of the terrain over a regularly spaced grid. Alternatives to such maps are point clouds, meshes, or three-dimensional grids, which provide a greater flexibility but have higher storage demands.

45.1 **Historical Overview** 1136
45.2 **Models for Indoors and Structured Environments** 1137
45.2.1 Occupancy Grids 1137
45.2.2 Line Maps 1138
45.2.3 Topological Maps 1140
45.2.4 Landmark-Based Maps 1141
45.3 **World and Terrain Models for Natural Environments** 1141
45.3.1 Elevation Grids 1141
45.3.2 3-D Grids and Point Sets 1143
45.3.3 Meshes 1145
45.3.4 Cost Maps 1145
45.3.5 Semantic Attributes 1146
45.3.6 Heterogeneous and Hierarchical Models 1148
45.4 **Dynamic Environments** 1149
45.5 **Summary and Further Reading** 1149
Video-References 1150
References 1150

The construction of models of the environment is crucial to the development of several applications of mobile robot systems. It is through these environment models that the robot can adapt its decisions to the current state of the world. The models are constructed from sensor data as the robot discovers its environment. There are three challenges in constructing environment models from sensor data. First, the models must be compact so that they can be used efficiently by other components of the system, such as path planners. Second, the models must be adapted to the task and to the type of environment. For example, modeling the environment as a set of planes is not relevant to a robot operating in natural terrain. In particular, this implies that a *universal* representation for mobile robots in not possible and that we must instead choose from an array of different approaches. Third, the representation must accommodate the uncertainty inherent to both sensor data and to the robot's state estimation system. The latter point is particular important since environment models typically accumulate sensor readings over substantial distances into a common reference frame. Drift in position estimates are unavoidable and must be accounted for in the model representation and in its construction.

45.1 Historical Overview

Historically, work in this area concentrated first on robots operating in indoor environments. In that case, the models take advantage of the fact that the world can be represented as vertical structures on reference ground planes. This simplification can be used for representing the world as a two-dimensional (2-D) grid. Uncertainty in the measurements and in the robot's pose can then be modeled by using probabilities of occupancy in the grid rather than binary occupied/empty flags. Another feature of indoor environments is that they are highly structured in that they contain primarily linear structures such as lines and planes. This observations led to a second class of representations based on collections of points, lines, and planes to represent the environment. Here again, a lot of attention was paid to representing the uncertainty on the relative poses of these geometric elements, including a considerable amount of work in the 1980s on using Kalman filters and other probabilistic techniques.

With continued progress on sensing (e.g., three-dimensional laser range scanners and stereo vision), and on mechanical and controls aspects of mobile robots systems, it became possible to develop mobile robot systems for operations in unstructured, natural terrain, motivated in part by planetary exploration and military applications. In these scenarios, it is no longer appropriate to project the data in a 2-D grid, and environments cannot be described adequately by a small vocabulary of geometric elements. Since, in most cases, it is still appropriate to assume (at least locally) that there is a reference ground plane, a natural representation is a two-and-a half-dimensional 2.5-D grid, in which each cell contains the elevation (and possibly other features) of the terrain at that location. Although they have been used extensively, the main challenges with such *elevation maps* are that they are not compact representations and that it is difficult to incorporate uncertainty in the representation. Accordingly, much of the research has focused on designing efficient data structures and algorithms for elevation maps, for example, hierarchical representations. Recently, the uncertainty challenge has been tackled by using probabilistic variants of elevation maps.

While elevation maps provide a natural representation for many types of natural terrains, they cannot represent environments with vertical or overhanging structures. This limitation has become more evident in recent years as applications of mobile robots are increasingly demanding operation in urban environment (in which building walls and other structures cannot be represented by elevation maps) and the use of aerial data, which involves overhanging structures such as tree canopies. This has led to the development of representations that are truly three-dimensional (3-D), such as point clouds, 3-D grids, and meshes. The two challenges described above apply here as well, except that the situation is more complicated because of the increased complexity introduced by the third dimension. Current research includes efficient computation over 3-D structures and probabilistic representations of 3-D data.

Because of the large volume of data involved in all of these representations, it is important to be able to group the data into larger chunks corresponding to semantically meaningful parts of the environment. This can be done at different levels, depending on the application the environment. At the lowest level, work in this area involves classifying the points into classes that are relevant to navigation tasks (e.g., distinguishing between vegetation and ground, extracting walls, tree surfaces, etc.). At an intermediate level of representation, this part of the work involves extracting parts of the environments that are considered landmarks of interest for the navigation task (e.g., roads). Finally, the highest representation level involves extracting and representing objects in the environments (e.g., natural obstacles, or specific objects such as cars for operation in urban environments).

All of these representations assume a static environment. In fact, many of the current applications of mobile robots call for operation in mixed environments in which the robot shares its environments with other moving agents, including other robots, people, and vehicles. Assuming that perception algorithms are able to detect and track individual moving objects in the environment, the challenge here is to insert this information into a representation that can be used by a planner. In this case, any one of the previous representations can be used as a foundation, but it needs to be extended to handle the temporal dimension by storing a history of the locations and states of the detected objects. Such representations include the trajectories of the detected objects and, in some cases, information on the future predicted trajectories of the objects.

45.2 Models for Indoors and Structured Environments

45.2.1 Occupancy Grids

Occupancy grid maps, which were introduced in the 1980s by *Moravec* and *Elfes* [45.1], are a popular, probabilistic approach to represent the environment. They are an approximative technique in which we calculate for each cell of a discrete grid the posterior probability that the corresponding area in the environment is occupied by an obstacle. The advantage of occupancy grid maps lies in the fact that they do not rely on any predefined features. Additionally, they offer a constant-time access to grid cells and provide the ability to represent unknown (unobserved) areas, which can be important, for example, in exploration tasks. Their disadvantage lies in potential discretization errors and the high memory requirements.

Throughout this section we assume that the map m consists of a discrete, two-dimensional grid of L cells denoted as $m_1, \ldots, m_L$. Given the sensory input $z_{1:t}$ obtained by the robot at the corresponding positions $x_{1:t}$, the occupancy grid mapping approach calculates a posterior probability $p(m \mid x_{1:t}, z_{1:t})$.

To keep the calculations tractable, the overall approach assumes that the individual cells of the grid are independent, i. e., that the following equation holds

$$p(m \mid x_{1:t}, z_{1:t}) = \prod_{l=1}^{L} p(m_l \mid x_{1:t}, z_{1:t}) \,. \tag{45.1}$$

Note that this assumption is rather strong. It basically states that any information about the occupancy of one cell does not tell us anything about its neighboring cells. In practice, we often find objects that are larger than the individual grid cells, like doors, cabinets, chairs etc. Thus, if we know that one cell is occupied, for each of its neighboring cells the probability raises that it is occupied. Despite this fact, occupancy grid maps have been successfully applied in numerous installations of mobile robots and have been proven to be a powerful tool that supports various navigation tasks such as localization and path planning.

Because of the independence assumption expressed by (45.1) we can concentrate on the estimation of the occupancy probability of the individual cells m_l in m. Under additional independence assumptions one can finally arrive at the following formula for calculating the occupancy probability $p(m_l \mid x_{1:t}, z_{1:t})$ that cell m_l is occupied given the prior probability $p(m_l \mid x_{1:t-1}, z_{1:t-1})$ and the new observation z_t observed at position x_t

$$\begin{aligned} &p(m_l \mid x_{1:t}, z_{1:t}) \\ &= \left[1 + \frac{(1-p(m_l \mid x_t, z_t))}{p(m_l \mid x_t, z_t)} \cdot \frac{p(m_l)}{(1-p(m_l))} \right. \\ &\quad \left. \times \frac{1-p(m_l \mid x_{1:t-1}, z_{1:t-1})}{p(m_l \mid x_{1:t-1}, z_{1:t-1})} \right]^{-1} . \end{aligned} \tag{45.2}$$

In practice, one often assumes that the prior $p(m_l)$ is 0.5 so that the second factor in the product becomes 1 and therefore vanishes from the equation.

Additionally, if we define

$$\mathrm{Odds}(x) = \frac{p(x)}{1-p(x)} \,, \tag{45.3}$$

the incremental updates can be computed using the following equation

$$\begin{aligned} &\mathrm{Odds}(m_l \mid x_{1:t}, z_{1:t}) \\ &= \mathrm{Odds}(m_l \mid x_t, z_t) \cdot \mathrm{Odds}(m_l)^{-1} \\ &\quad \times \mathrm{Odds}(m_l \mid x_{1:t-1}, z_{1:t-1}) \,. \end{aligned} \tag{45.4}$$

To recover the occupancy probability from the Odds representation given in (45.4), one can use the following formula, which can easily be derived using (45.3)

$$p(x) = \frac{\mathrm{Odds}(x)}{1 + \mathrm{Odds}(x)} \,. \tag{45.5}$$

It remains to describe how to compute the occupancy probability $p(m_l \mid x_t, z_t)$ of a grid cell given a *single* observation z_t and the corresponding pose x_t of the robot. This quantity strongly depends on the sensor of the robot and has to be defined individually for each type of sensor. Additionally, the parameters of these models have to be adapted according to the properties of each sensor. Let us assume that the function $\mathrm{dist}(x_t, m_l)$ refers to the distance between the sensor at pose x_t and the center of the cell m_l. Let us first assume that we only need to consider the optical axis of the sensor cone as is, for example, the case for laser beams sent out by a laser range finder. Then, $p(m_l \mid x_t, z_t)$ can be formulated as

$$p(m_l \mid x_t, z_t) = \begin{cases} p_{\text{prior}}, & z_t \text{ is a maximum range reading} \\ p_{\text{prior}}, & m_l \text{ is not covered by } z_t \\ p_{\text{occ}}, & |z_t - \mathrm{dist}(x_t, m_l)| < r/2 \\ p_{\text{free}}, & z_t \geq \mathrm{dist}(x_t, m_l) \end{cases} , \tag{45.6}$$

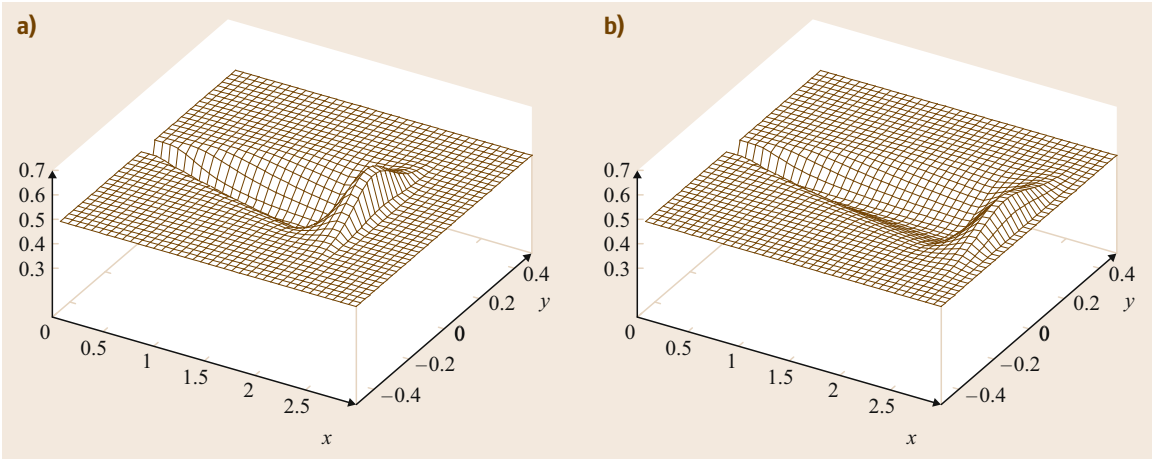


Fig.45.1a,b Occupancy probability introduced by a single ultrasound measurement of **(a)** $z = 2.0\,\mathrm{m}$ and **(b)** $z = 2.5\,\mathrm{m}$ (after [45.2])

where r is the resolution of the grid map. Obviously, it must hold $0 \leq p_{\mathrm{free}} < p_{\mathrm{prior}} < p_{\mathrm{occ}} \leq 1$.

If a sonar sensor is used, the sensor model is slightly more complicated, since the sensor is not a beam sensor and the observations are more noisy than the ones of a laser range finder. To deal with these properties, one can use a mixture of three functions to express the model. First, the influence of an observation (which is represented by the difference between p_{prior} and p_{occ} as well as between p_{prior} and p_{free}) decreases with the measured distance. Second, the proximity information of a sonar is substantially affected by noise. This can be considered by employing a piecewise linear function to model a smooth transition from p_{free} to p_{occ}. Finally, the sonar sensor should not be modeled as a beam sensor, since it sends out a conic signal. The accuracy of an observation decreases with the angular distance between the cell under consideration and the optical axis of the observation. This is expressed by the derivation from the prior and is typically modeled using a Gaussian with zero mean. Therefore, it is maximal along the optical axis and decreases the bigger the angular distance form the optical axis is [45.2].

Two examples of resulting models are depicted in Fig. 45.1, which shows two three-dimensional plots of the resulting occupancy probabilities for a measurement of 2 m (Fig. 45.1a) and 2.5 m (Fig. 45.1b). In this figure, the optical axis of the sensor cone was identical with the x-axis and the sensor was placed in the origin of the coordinate frame. As can be seen, the occupancy probability is high for cells whose distance to x_t is close to z_t. It decreases for cells with shorter distance than z_t as well as with increasing values of the angular distance.

Figure 45.2 depicts the mapping process for a sequence of observations recorded with an iRobot B21r robot. The first row shows how a map was built from a sequence of previous ultrasound scans. Afterwards the robot perceived a series of 18 ultrasound scans, each consisting of 24 measurements. The occupancy probabilities for these 18 scans are depicted in rows 2–7. The occupancy probability grid obtained by integrating the individual observations into the map is shown in the last row of this figure. As can be seen, the belief converges to a representation of the corridor structure in which the scans where recorded. A typical resulting map obtained for an indoor environment is depicted in Fig. 45.3.

45.2.2 Line Maps

The representation of the environment by line models is a popular alternative to the grid-based approximations described above. Line models have several advantages over these nonparametric representations. They require substantially less memory than grids and therefore scale better with the size of the environment. They furthermore are more accurate since they do not suffer from discretization problems. In this section, we consider the problem of calculating lines that provide the approximation of a set of range points by a line. If the data points are given by n pairs (x_i, y_i) of Cartesian coordinates, the line that minimizes the squared distances to all points can be calculated in closed form according to

$$\tan 2\phi = \frac{-2\sum_i (\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i [(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]}\,, \tag{45.7}$$

$$r = \bar{x}\cos\phi + \bar{y}\sin\phi\,, \tag{45.8}$$

where $\bar{x} = \frac{1}{n}\sum_i x_i$ and $\bar{y} = \frac{1}{n}\sum_i y_i$. In these equations, r is the normal distance of the line from the origin and ϕ is angle of the normal.

Unfortunately, there is no closed-form solution for situations in which the data points are generated by multiple linear structures. In such a situation two problems arise. First one has to answer the question how many lines there are, and second one has to solve a data association problem, which is to find the assignment of the data points to the individual lines. Once the number of lines and the associations are known, we can apply (45.7) and (45.8) to calculate the individual line parameters.

There is a popular approach to solve this problem for range scans with multiple linear structures. This approach, which goes back to the work of *Douglas* and *Peucker* [45.3], is also known as the *split-and-merge* algorithm. Its key idea is to recursively subdivide the point set into subsets that can be more accurately approximated by a line. The approach starts with the line calculated from all data points and determines the data point with maximum distance from this line. If this distance is below a given threshold, the algorithm terminates and provides the fitted line as output. Otherwise, it calculates the point with the largest distance from the line that connects the first and the last point. This point is the so-called splitting point. It then divides the point set into two subsets, one containing all points from the first up to the splitting point and one containing all points after the splitting point. The algorithm is then recursively applied to the two subsets. Figure 45.4 shows a visualization of the procedure.

Whereas the split-and-merge algorithm is quite effective and efficient, it is not guaranteed that the resulting model is actually optimal, i. e., is the model that minimizes the squared distances of all data points. One approach that is able to find such a model is based on the expectation maximization (EM) algorithm, which in the application described here can be regarded as a variant of the fuzzy k-means clustering algorithm. Let us suppose that the number m of lines in the model θ is known. Let us furthermore suppose that the likelihood of a data point $z = (x, y)$ given the model θ consisting of the line set $\{\theta_1, \ldots, \theta_m\}$ is defined as

$$p(z \mid \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\frac{d(z, \theta_k)^2}{\sigma^2}\right), \qquad (45.9)$$

where σ is the standard deviation of the measurement noise and θ_k is the line for which Euclidean distance $d(z, \theta_k)$ to z is minimal.

The goal of the EM algorithm is to generate an iterated sequence of models of increased likelihood. To achieve this, one introduces so-called correspondence variables $c_{ij} \in \{0, 1\}$ that specify to which linear component of the model each point belongs. Since the correct values of these assignment variables are unknown, one estimates a posterior about their values. Let θ_j be a component of the model and z_i be a measurement. Then the expectation about c_{ij}, i. e., that measurement i belongs



Fig. 45.2 Incremental mapping in a corridor environment. The *upper left* image shows the initial map and the *lower* one shows the resulting map. The maps *in between* are the local maps built from the individual ultrasound scans perceived by the robot (after [45.2])



Fig. 45.3 Occupancy grid map obtained from ultrasound data (after [45.2])

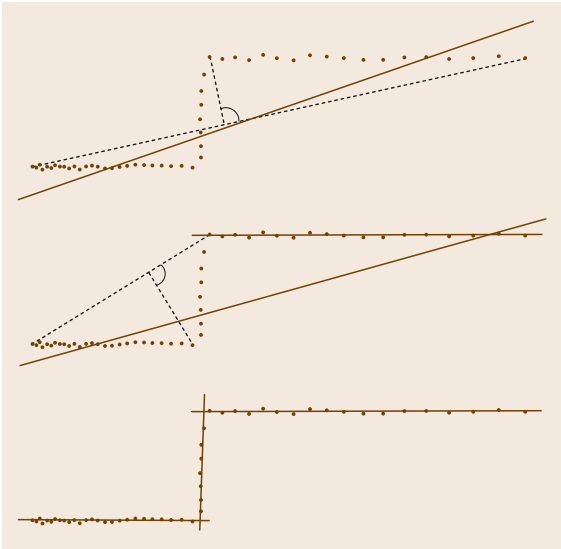


Fig. 45.4 Split-and-merge algorithm applied to a point set

to line j is computed in the E-step as

$$E[c_{ij} \mid \theta_j, z_i] = p(c_{ij} \mid \theta_j, z_i) \,, \tag{45.10}$$

$$= \alpha p(z_i \mid c_{ij}, \theta_j) p(c_{ij} \mid \theta_j) \,, \tag{45.11}$$

$$= \alpha' p(z_i \mid \theta_j) \,. \tag{45.12}$$

In the M-step, the algorithm then computes the parameters of the model by taking into account the expectations computed in the E-step

$$\theta_j^* = \arg\min_{\theta_j'} \sum_i \sum_j E[c_{ij} \mid \theta_j, z_i] d^2(z_i, \theta_j') \,. \tag{45.13}$$

Given a fixed variance σ for all data points in z we can determine the most likely model in closed form according to the following equations, which are probabilistic variants of (45.7) and (45.8) that take into account the data association uncertainty given by the expectations calculated in the E-step

$$\tan 2\phi_j = \frac{-2 \sum_i E[c_{ij}|\theta_j, z_i](\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i E[c_{ij}|\theta_j, z_i][(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]} \,, \tag{45.14}$$

$$r_j = \bar{x} \cos\phi_j + \bar{y} \sin\phi_j \,. \tag{45.15}$$

Here $\bar{x}$ and $\bar{y}$ are computed as

$$\bar{x} = \frac{\sum_i E[c_{ij}|\theta_j, z_i] x_i}{\sum_i E[c_{ij}|\theta_j, z_i]} \,, \quad \bar{y} = \frac{\sum_i E[c_{ij}|\theta_j, z_i] y_i}{\sum_i E[c_{ij}|\theta_j, z_i]} \,. \tag{45.16}$$

Figure 45.5 depicts a line map extracted from 311 823 data points using the EM-based approach. In this example, the model consists of 94 lines. One approach to determine the optimal number of lines is to utilize the Bayesian information criterion [45.4].



Fig. 45.5 Line map consisting of 94 lines generated for 311 823 range points generated with the EM-based approach (after [45.4])

45.2.3 Topological Maps

In contrast to the previously discussed representations, which mainly focus on the geometric structure of the environment, topological representations have also received significant attraction. One of the pioneering approaches to topological mapping, which has been presented in 1988, is the work by *Kuipers* and *Byun* [45.7]. In this approach, the environment is represented by a graph-like structure, in which the nodes are locally distinguishable places and the nodes are travel edges along which the robot can move between the places (VIDEO 270). Here, distinctive places are identified according to the distance to nearby objects. The distinctive places were been defined by *Choset* and *Nagatani* [45.8] as the meet-points in the generalized Voronoi diagrams, i. e., the points with an out-degree of three or more. A generalized Voronoi diagram is the set of points equidistant from the closest two or more obstacle boundaries. Generalized Voronoi diagrams are a very popular representation as they can be considered as road-maps with a high correspondence to the topological structure of the environment. They

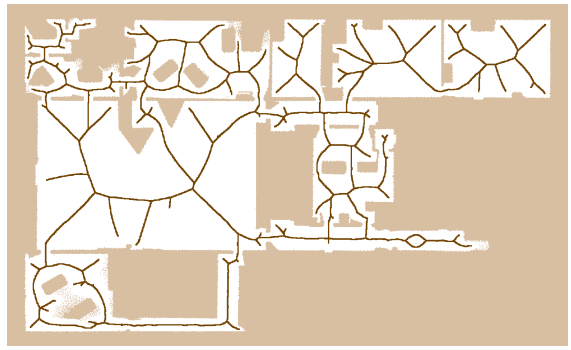


Fig. 45.6 Example of a generalized Voronoi graph (after [45.5])



Fig. 45.7 (a) Mobile robot mapping a set of stone rocks. The image (b) depicts the path and the estimated landmark positions. The manually determined positions of the rocks are marked by *circles* (after [45.6])

have been used extensively for path planning. To plan a path from a starting position to a goal point in the environment, all the robot has to do is to first plan a path to the generalized Voronoi graph, than along the generalized Voronoi graph, and then from the generalized Voronoi graph to the goal point [45.8]. Figure 45.6 shows an example generalized Voronoi graph for an indoor environment. Note that in this figure only those parts of the graph that can be traversed by the robot without colliding with objects are displayed.

45.2.4 Landmark-Based Maps

For environments with locally distinguishable features, landmark-based maps have been extensively used. If we assume that the position of the robot is always known, it simply remains to maintain an estimate about the positions of the individual landmarks over time. In a planar environment, m is represented by K two-dimensional Gaussians with mean μ_k and covariance Σ_k, one for each landmark. If a linearized version of the perception problem is given, the individual Gaussians can be updated using the equations for the extended Kalman filter. Note that, compared to the use of the extended Kalman filter EKF for landmark-based simultaneous localization and mapping (SLAM), where we needed a $2K+3$-dimensional state vector (3 dimensions for the position of the robot and $2K$ dimensions for the positions of the individual landmarks), we only need K two-dimensional Gaussians to represent the entire map since the robot pose is known. This property, for example, has been utilized in the FastSLAM algorithm [45.6]. Over the last few years, the graph-based SLAM paradigm has become a popular approach for estimating landmark maps due to their ability of relinearizing the motion and the measurement functions (for more details see Chap. 46).

Figure 45.7a shows a robot equipped with a SICK laser range finder that maps the position of rocks. The right image depicts the path of the vehicle as well as the manually determined and automatically estimated landmark positions.

45.3 World and Terrain Models for Natural Environments

From a survey of environment modeling [45.9] with an emphasis on probabilistic techniques for indoor environments, a taxonomy can be created along several axis: metric versus topological versus semantic, robot-centric versus world-centric, or application-based. We choose to first review purely geometric models (elevation grid, 3-D grid, meshes), then geometric models with low-level attributes (cost maps), and then models with richer semantic attributes, to finish with heterogeneous and hierarchical models.

45.3.1 Elevation Grids

Assuming that the terrain can be represented as a function $h = f(x, y)$, where x and y are the coordinates of a reference place and h is the corresponding elevation. A natural representation is a digital elevation map which stores the value of h at discrete locations (x_i, y_i). Because they are simple data structures and can be generated from sensor data in a relatively straightforward manner, elevation maps have been used extensively for mobile robots operating in natural environments with no vertical surfaces or overhangs (e.g., for planetary exploration scenarios [45.10]). Several issues need to be addressed when using elevation maps for mobile robots.

A regularly sampled grid is appropriate when the sensor data is roughly uniformly distributed on the reference plane. This is the case, for example, with aerial data. For ground robots, however, the distribution of the data on the reference ground plane varies dramatically

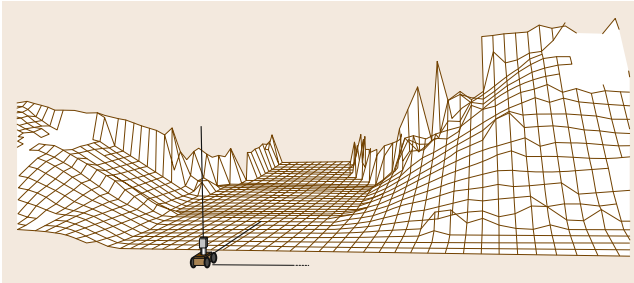


Fig. 45.8 Example elevation map built by accumulating 3-D data from a range sensor

because of the small incidence angles with the reference plane. This can be addressed by using variable-size cells instead of regularly sampled cells. In that case, the distribution of the cells on the reference plane is designed to approximate the distribution of points that would be measured by the sensor on this reference plane. Such nonuniform representations are useful primarily in cases in which the map is referenced to the current position of the robot [45.11] (Fig. 45.9). Frequent resampling is needed if the map is expressed in a global reference frame.

Irrespective of the sampling scheme used to construct the reference grid, the density of data in the grid varies due to local self-occlusions of the terrain surface and to mismatches between the resolution of the projected sensor data and the resolution of the grid. This can be problematic because, from the point of view of a planner, the elevation map would appear as a scattering of cells with elevation data and cells with no evidence (Fig. 45.8). Various schemes have been proposed to remedy this problem. The basic idea is to estimate the elevation values of the empty cells by interpolating across the cells with know elevation values. This has to be done with great care to avoid filling up regions that correspond to parts of the environment occluded by the terrain (*range shadows*) in which elevation data should not be inferred. Such mistakes can be catastrophic since a planner may generate paths through areas that are entirely unknown. One general approach to this problem is to use surface interpolation techniques including a term that allows for discontinuities in the resulting surfaces [45.12]. An alternative approach is to use the visibility constraints induced by the known geometry of the sensor in order to estimate plausible values of elevation at each cell [45.13, 14].

A difficulty in interpolating techniques is to explicitly take into account the sensor uncertainty. In particular, it is difficult to account for the varying resolution of the sensor as a function of range in a fixed-resolution grid. An alternative is to use multiple grids at different resolutions. The appropriate resolution to be used at a particular point (x, y) can be decided based on the range from (x, y) to the closest sensor location to retrieve the value at (x, y) from the appropriate map for that resolution. Generally, coarser map resolutions are used for longer ranges. This approach eliminates the gaps in the map due to undersampling at long range from the sensor by using the optimal resolution based on sensor geometry [45.15].

Uncertainty in the sensor measurement is expressed most naturally with respect to the sensor frame, for example, by expressing the uncertainty in the direction of the measurement. As a result, converting such uncertainty models to elevation maps is difficult because a distribution in the direction of the measurement maps to a distribution in the reference plane, not a distribution on the elevation value at a particular grid point. When updating a cell based on sensory input, one has to take into account that the uncertainty in a measurement increases with the distance measured due to errors in the tilting angle. A popular approach to estimate the height h at location (x, y) is to use a Kalman filter. If we assume that σ is the standard deviation of the current measurement h in the vertical direction in (x, y) and σ_{t-1} is the standard deviation of the current estimate or h_{t-1}, we can apply the following equations to

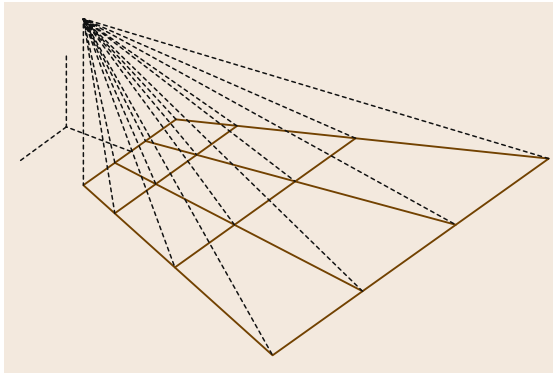


Fig. 45.9 Example of a robot-centric map with variable size (after [45.11])

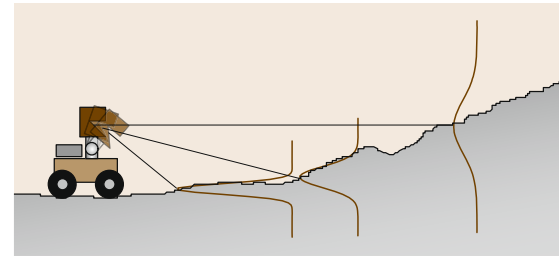


Fig. 45.10 The standard deviation of a height measurements can be modeled to depend linearly on the distance of the grid cell to the sensor

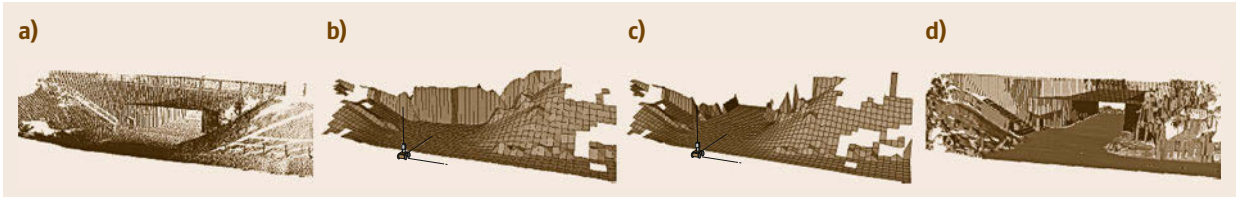


Fig.45.11a–d Different versions of digital elevation maps (after [45.16]): **(a)** scan (point set) of a bridge, **(b)** a standard elevation map computed from this data set, **(c)** an extended elevation map, which correctly represents the underpass under the bridge, and **(d)** a multilevel surface map that correctly represents the height of the vertical objects

obtain the new estimate h_t with standard deviation σ_t,

$$h_t = \frac{\sigma^2 h_{t-1} + \sigma_{t-1}^2 h}{\sigma_{t-1}^2 + \sigma^2} , \tag{45.17}$$

$$\sigma_t^2 = \frac{\sigma_{t-1}^2 \sigma^2}{\sigma_{t-1}^2 + \sigma^2} . \tag{45.18}$$

One possible solution to deal with varying uncertainties is to use a model in which the standard deviation of the height of a measurement increases linearly with the length of the range measurement, as indicated in Fig. 45.10.

Vegetation cover is another source of error in the recovery of terrain elevation by totally or partially occluding the ground surface from the vehicle sensors. Online learning techniques can address this issue by predicting ground elevation ahead of the vehicle based on past appearance observations of similar terrain and vehicle state over the terrain [45.17].

Intermediate representations between 2-D elevation maps and full 3-D representations, addressed below, are extended elevation maps [45.18] and so-called multilevel surface maps [45.16]. Such approaches are specifically useful in structured terrain with vertical objects or overhangs such as bridges (Fig. 45.11).

45.3.2 3-D Grids and Point Sets

Elevation maps as described above assume a reference direction. In many cases, that assumption is violated. An alternative is to represent the data directly in 3-D without projecting it on a reference 2-D plane. The advantage of doing this is that all the sensor data can be preserved in its original distribution and that there is no restriction on the geometry of the environment. Figure. 45.12 shows an example representation using 3-D points. The drawbacks of SLAM systems operating on such point clouds [45.19–21] is that neither free space nor unknown areas can be modeled and that sensor noise and dynamic objects cannot be dealt with directly. Another problem is that very large sets of 3-D points are difficult to handle efficiently. The point cloud library [45.22] addresses this problem by providing an octree-based implementation to store and address large point clouds in an efficient manner.

Data structures based on dynamic 3-D grids can also be used for this purpose [45.26]. An interesting characteristic of these representations is that they enable the computation of cost evaluated from the true local 3-D distribution of the data (as opposed to costs

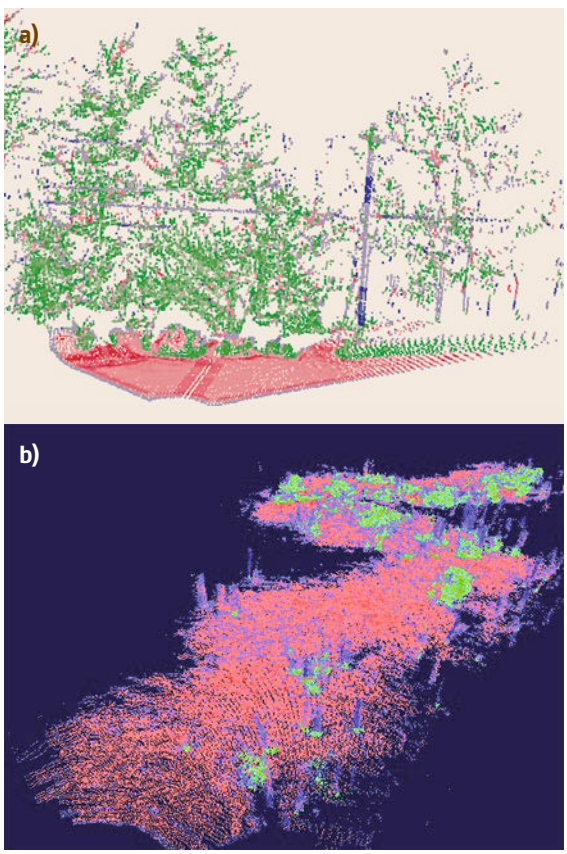


Fig.45.12a,b Example of mapping and classification of 3-D point sets: **(a)** 3-D data and classification result (*green* = vegetation, *red* = surfaces, *blue* = lines; the saturation of the color is proportional to the confidence in the classification result); **(b)** map accumulated over a large number of scans

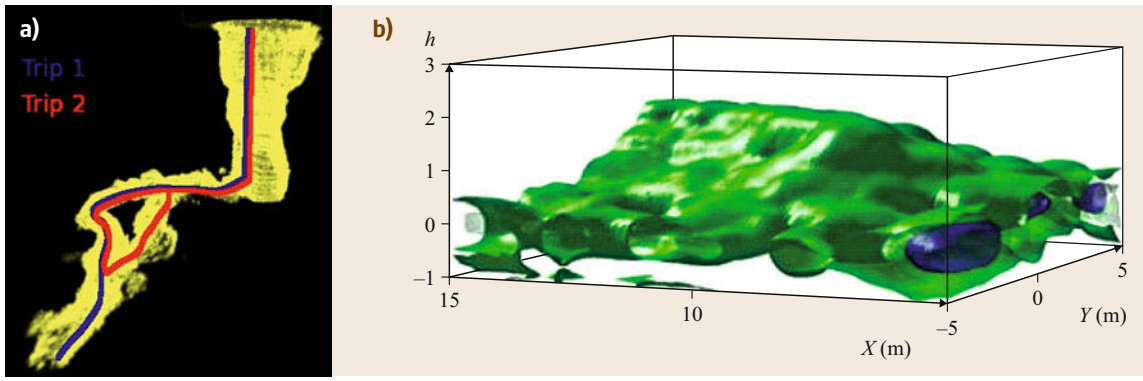


Fig.45.13a,b Examples of volumetric maps for a natural environment. Whereas **(a)** shows a map of an underwater cave (after [45.23]), **(b)** depicts a terrain map obtained from a radar (after [45.24])

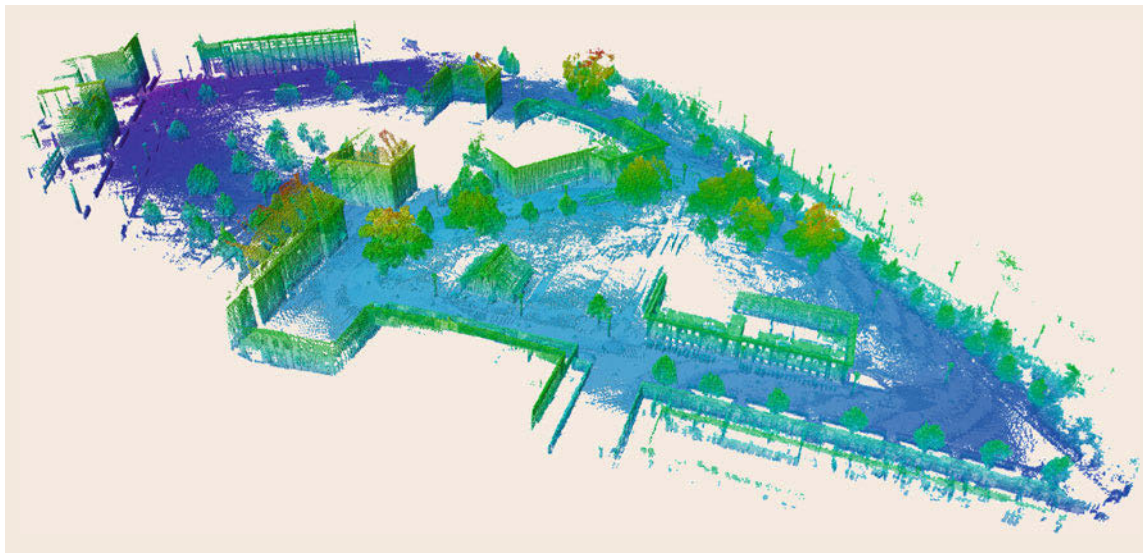


Fig. 45.14 Resulting octree-based representation of an outdoor environment at 0.2 m resolution (size of the scene: 292 m × 167 m × 28 m) (after [45.25]). For clarity, only occupied volumes are shown with height visualized by a color (*gray scale*) coding

computed from a local surface $z = f(x, y)$). This is important, for example, in environments with vegetation scatter, which cannot be modeled as a surface.

Extensions of occupancy grids to dense 3-D occupancy grids have been used successfully to model natural environments using a radar [45.24]. For more efficiently maintaining 3-D representations of even large-scale environments, also tree-based representations can been employed. In particular, octree-based 3-D representations have been employed, for example, to map an underwater cave from several sonars mounted on an underwater vehicle [45.23]. Example maps for these methods are presented in Fig. 45.13.

The OctoMap approach [45.27] uses octrees and supports a compact memory representation, multiresolution queries, and probabilistic occupancy estimates. The volumetric representation of space allows for explicitly representing free space and unknown areas. To exploit hierarchical dependencies in the environment it can be extended to maintain a collection of submaps in a tree-structure, where each node represents a subspace of the environment [45.28]. Figure. 45.14 and VIDEO 79 show an example outdoor map acquired from laser data and built by OctoMap. The OctoMap framework has been used successfully for various tasks including autonomous navigation with air vehicles [45.29], humanoid navigation in multilevel indoor environments [45.30], and mobile manipulation in cluttered environments [45.31, 32] where volumetric information is necessary to plan paths.

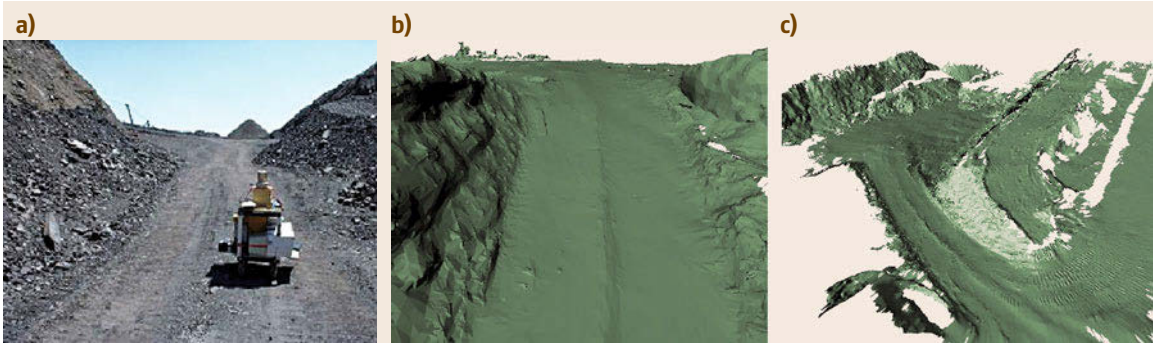


Fig.45.15a–c Two views (**b–c**) of the mesh representation of a map (**a**) obtained by scanning an environment.The meshes are reduced by a factor of 10 from the initial data

In the approaches mentioned above, the data sets are accumulated, sometimes probabilistically, into discrete environment volumes. A different approach is to represent the environment with discrete sensor samples integrated probabilistically into a metric map [45.33]. Such an approach is different from occupancy grids in several aspects: the storage requirement and resolution is adaptive instead of fixed and the potential for extensibility and scalability is higher.

45.3.3 Meshes

As described above, elevation maps are compact and easy to implement but they are restricted to a particular class of terrain; at the other extreme, using 3-D representations directly is more general but it is also more expensive computationally, and it does not represent explicitly surface continuity. A compromise is to represent the map by a mesh. This approach is attractive because it can, in principle, represent any combination of surfaces. It is also a compact representation in that, even though the size of the mesh may be initially very large, efficient mesh simplification algorithms exist in the literature [45.34] and can be used to reduce the map to a small number of vertices.

The key issue with meshes is that extraction of the correct surfaces from raw data can be difficult in complex environments. In practice, the data may be corrupted by sensor noise and by random clutter from other sources, such as vegetation, which cannot be represented as a continuous surface. In addition, it is necessary to precisely detect discontinuities in the data so that disconnected pieces of surface do not become accidentally linked in the mesh formation process [45.18]. Figure 45.15 shows typical mesh representations of an urban environment.

In some applications, the objective is to produce terrain models not to support autonomous navigation but to produce a model to be visualized to a human. City planning is a typical application. Examples include producing 3-D texture models of urban environments using videos from a camera mounted on a moving vehicle equipped with an inertial navigation system [45.35] (VIDEO 269), using two laser scanners in push-broom mode, one for mapping and the other for localization [45.36], with a camera coregistered with the mapping laser, and collecting high-resolution laser data and imagery to produce geometric and photometric correct 3-D models of buildings [45.37]. Figure 45.16 depicts examples of textured meshes representing urban terrain.

45.3.4 Cost Maps

The most direct use of elevation maps is to compute traversability costs at each cell of the grid. The costs are computed by comparing the local terrain shape with a kinematic model of the robot. Several approaches have been proposed for computing the costs depending on the exact vehicle model; for example, a cost can be computed by considering the local slope and 3-D texture of the terrain [45.38]. More involved models take into account a detailed model of the robot dynamics. In that case, different costs are generated for different possible robot speeds and path curvatures. The cost grid is used in a minimum-cost planner (VIDEO 271). Since the terrain map is updated continuously as the robot traverses the environment, it is also necessary to update the costs continuously as new data arrives. Consequently, it is important that the planner is able to support changing costs without needing to process the entire grid every time new data is inserted.

An example of such a combination of grid representation and dynamic planner is the D* system, which uses a version of A* that fully supports dynamic grid updates [45.39–41]. Whenever a grid cell (or a group of grid cells) is updated, the planner makes minimal updates to its internal representation so that the optimal path can be updated quickly.



Fig.45.16a–c Examples of urban terrain modeled as a textured mesh **(a)** (after [45.35]), **(b)** (after [45.37]), and **(c)** (after [45.36])

Defining the exact relation between the costs and the elevation values stored in the grid can be quite difficult. In fact, beyond the limiting cases of terrain with large elevation gradient or large slope, there is little guidance as to why a particular part of the terrain may be more traversable than another as it depends critically on the exact configuration of the robot. For this reason, recent work has focused on deriving cost maps directly from observations, rather than by using handcrafted algorithms. One approach involves learning the best weights to combine a set of predefined costs. Another approach uses costs computed from other sources for inferring how to determine the costs on the terrain currently seen by the robot; for example, data from overhead imagery can be used to predict the traversability costs that should be used for a ground robot [45.42] (Fig. 45.17) or for learning to map local elevation distributions to costs values by analyzing actual robot trajectories through the terrain. Similar online learning approaches are used to predict terrain roughness [45.43], terrain slippage [45.44], or traversability in general [45.45, 46] to be used in a cost map.

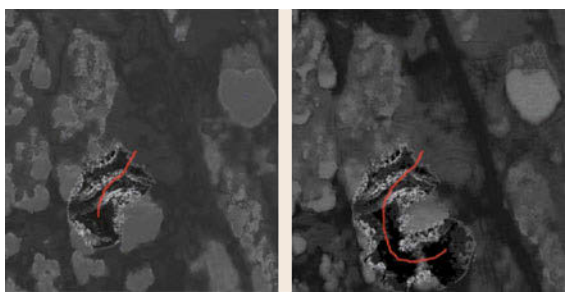


Fig. 45.17 Example of online learning of cost (after [45.42]). The vehicle trajectory is represented in *red*. As the vehicle progresses, note the change in cost over all the map. The *darker* the appearance, the *lower* the cost

45.3.5 Semantic Attributes

The representations described above are concerned only with storing the data in a way that is compact and that can be used to estimate drivability costs. Often it is necessary to reason with high-level knowledge about the environment, e.g., the location and types of objects around the robot or the type of terrain (vegetation, mud, wall) in the environment. For convenience we will refer to this type of information as *semantic attributes* attached to different parts of the environment. There are several different ways to approach the problem of generating and representing semantic attributes. One possible direction is to approach the problem as

one of extracting *landmarks* from the environment. As for their indoor counterparts, outdoor environments can also be modeled using landmarks. Landmarks are defined broadly as scene elements easy to detect, salient in their surroundings, and easy to recognize. They can be specific objects, such as rocks [45.47], tree trunks in forest [45.48], or locations with a distinct appearance signature in 2-D or 3-D, for urban terrain [45.49, 50] or natural environments [45.51]. Landmarks can be used to produce topological representations of the environment or used in conjunction with metric maps. Recent efforts involve learning a compact representation of environments without defining a priori what a landmark should be [45.52, 53] or using statistical learning techniques [45.54] to extract new features from images. Nonlinear dimension-reduction techniques are used to obtain the representation and the results were demonstrated onboard ground and aerial vehicles (Fig. 45.18).

Another view of the problem is as a *classification and grouping* problem. Indeed, the map representations described in the earlier sections are low-level representations in that they do not attempt to group the data points into larger structures that are coherent with respect to geometry, terrain type, or semantic content. In practice, one would like to abstract the data into larger units that can be used by a planner or transmitted to another robot or an operator; for example, in an urban environment, one would like to group the parts of the data that belong to planes corresponding to pieces of walls. One such example is presented in Fig. 45.19.

One approach is to group 3-D points into components based on low-level classification and feature detection [45.58]. The shape of these component is analyzed further to discriminate between various natural object components (tree trunks, branches) and man-made obstacle (wires). Geometric primitives are then fitted to the components (mesh for the ground surface, ball tree for the vegetation, and cylinder to branches) to produce a compact high-level geometric description of the terrain.

Local terrain classification can be achieved by computing local features from the map and classifying each element of the map in different classes of terrain. This can be done from an elevation map, in which case the *elements* are the cells of the elevation map, or from a point cloud representation in which case the elements are 3-D locations. Given a location in the map x, a feature vector is $\boldsymbol{V}(x)$ is computed in a neighborhood $N(x)$ of x and a classifier $f(V)$ returns the type of terrain at x. Features that have been used in the past are statistics of the distribution of the map data around each element, such as the slope and distribution of elevation [45.59], and the second-order moment of the distribution of the 3-D points in a neighborhood [45.58]. The terrain classes depend on the application. The most direct classification approach uses a binary classifier that separates the terrain into obstacle regions and traversable regions. More involved classifiers segment the map into more classes such as vegetation, solid surfaces, and linear structures [45.13, 58, 60]; an example is shown in Fig. 45.12. In some cases, it is possible to store, with each data element x in the map, the direction in which a measurement was taken for this element. In this case, it is possible to refine the classification by rea-

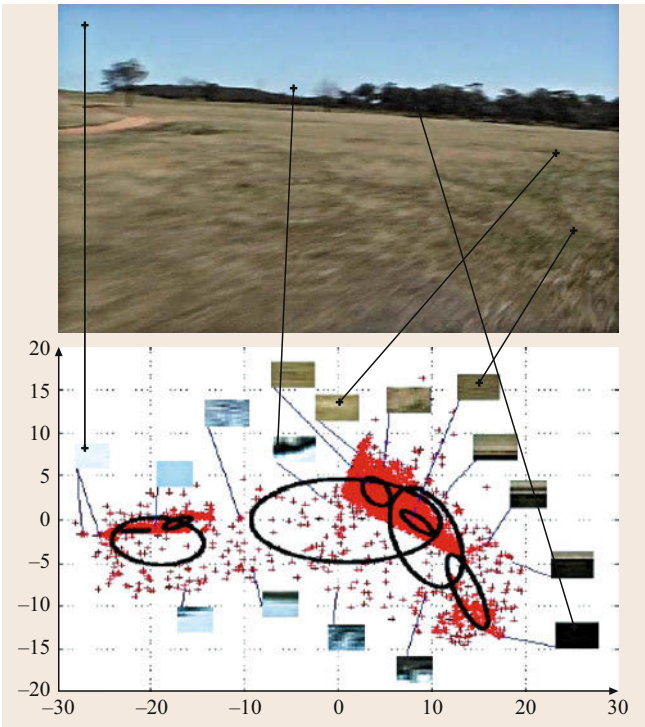


Fig. 45.18 Sample image and low-dimensional embedding of randomly sampled high-dimensional image patches (after [45.53])

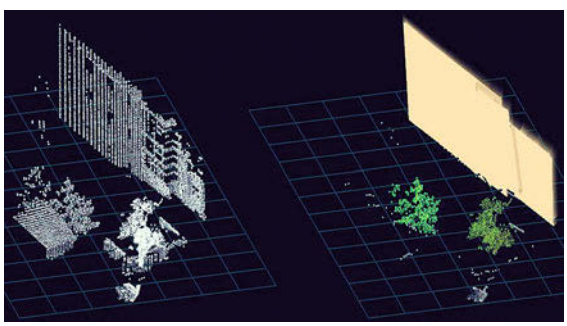


Fig. 45.19 Terrain classification and extraction of geometric features.Whereas the 3-D data is shown on the *left*, the feature map with extracted planes and vegetation regions is shown on the *right* (after [45.55])

Avidbots Ex1016
Page 1175 of 2154

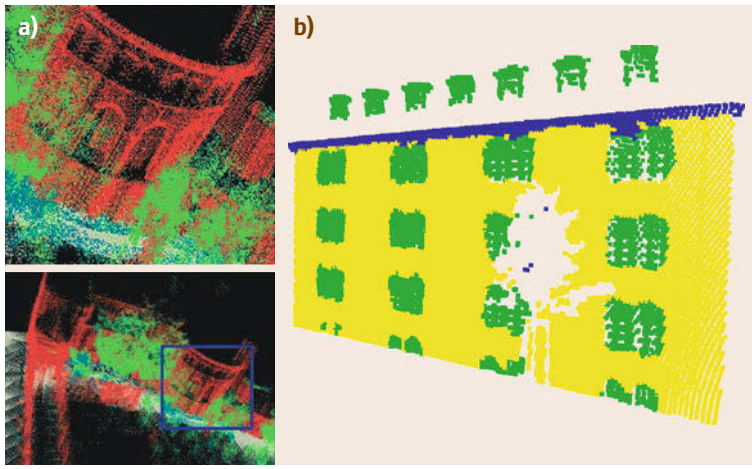


Fig. 45.20 (a) Structure learning for terrain classification (after [45.56]) and (b) building features extraction (after [45.57])

soning about the intersection of the measurement ray $d(x)$ with the rest of map; for example, this type of geometric reasoning [45.61] has been exploited to recover the load-bearing surface obscured by vegetation [45.51, 62, 63] and for extracting *negative* obstacles (such as ditches) [45.64]. In all of these cases, the classification information cannot be recovered directly from local statistics and must be inferred from longer-range geometric reasoning.

This class of approaches to local feature classification is similar to approaches for extracting features from images and it suffers from similar limitations. Specifically, these representations are sensitive to the choice of neighborhood used for computing the features. If it is too large, information over a large area is averaged out, leading to poor classification performance. If it is too small, there is not enough information in the neighborhood for reliable classification. The situation is complicated by the fact that the resolution of the map, or more precisely the density of data points in the map, may vary drastically as the distance from the sensor changes. This problem is addressed by using different neighborhood sizes at different locations in the map, or by tuning the classifier differently depending on the map location, typically based on distance from the sensor [45.15, 65]. A second problem is to generate the classifier f. This can be done by using a physical model that predicts the statistics of the local data distribution in the map assuming different terrain types [45.66]. This is generally difficult and a preferred approach is to train the classifier on training data.

This level of classification provides information about the local type of terrain, which can be used in planning, but it does not extract the extended geometric structures that may exist in the environment (Fig. 45.19). Geometric structures such as planar patches can in principle be extracted by using techniques similar to the ones described in the context of indoor environments, such as EM, with the added difficulty that there is a larger amount of clutter that complicates the extraction of the patches. Robust techniques that can handle this level of clutter are typically used for extracting the planes [45.67, 68].

Classification based on local attributes can be improved significantly by taking into account contextual information; for example, hidden Markov models have been successfully used for laser data analysis to determine terrain traversability [45.69]. Another structure learning approach based on Markov random fields in conjunction with margin-maximization criterion has been demonstrated for a wide class of applications, including 3-D terrain classification and object segmentation [45.56] or building structures extraction [45.57] (Fig. 45.20). Finally there are approaches to detect, select, model, and recognize natural landmarks automatically for robot localization and environment mapping [45.47].

45.3.6 Heterogeneous and Hierarchical Models

For long-range navigation, an autonomous vehicle must perform numerous tasks, including absolute and relative localization, path planning, and reactive obstacle avoidance. In addition, it must perform some tasks to fulfill the mission requirements, detecting and modeling objects, for example. To achieve this, a hierarchical framework has been presented that accounts for the different scales and granularities of the representation needed [45.59]. Furthermore, models built from heterogeneous imagery sources (overhead, descent, and ground) have been used to produce 3-D multiresolution terrain models that support Mars exploration [45.70]. The hybrid metric map [45.71], enhances feature maps

Avidbots Ex1016
Page 1176 of 2154

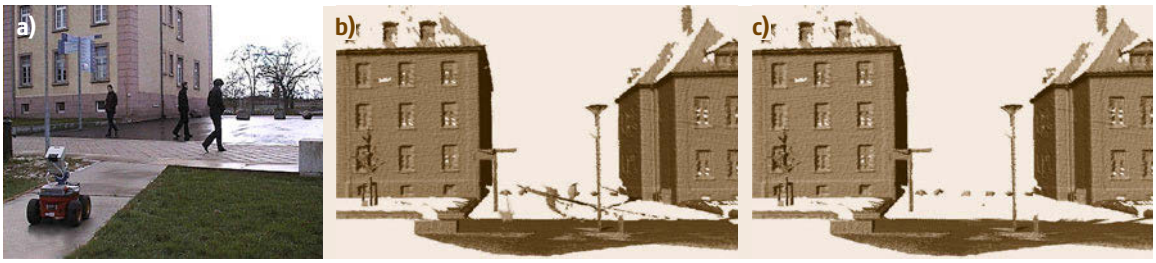


Fig. 45.21 (a) A mobile robot acquiring a three-dimensional scan of an urban scene. (b) The people in the scene cause spurious data points in the resulting meshes. (c) The same scene after filtering people

with metric maps to provide a dense but compact environment representation. Another representation extracts landmarks based on laser/imagery appearance as well as a metric map [45.72].

45.4 Dynamic Environments

The majority of techniques for mapping have been developed for static environments. Certain approaches like occupancy grids or elevation maps can in principle deal with dynamic environments in which objects move. Their drawback lies in the fact that the time to unlearn that a cell is free or that the elevation has changed can require as many observations as the robot received with the same area being occupied or with a different elevation. To resolve this problem, several alternative approaches have been proposed in the past. One very popular technique is to track moving objects using feature-based tracking algorithms [45.73, 74]. Such approaches are especially useful when the type of the dynamic objects is known in advance. They have been successfully applied in the context of learning three-dimensional city maps from range data. Figures 45.21 and 45.22 show applications in which dynamic objects were removed from the 3-D data acquired with mobile robots in urban scenes. Alternative approaches to such tracking techniques include those that learn maps on different time scales [45.75], that explicitly learn different states of dynamic environments [45.76, 77], or that only map the static aspects [45.78].

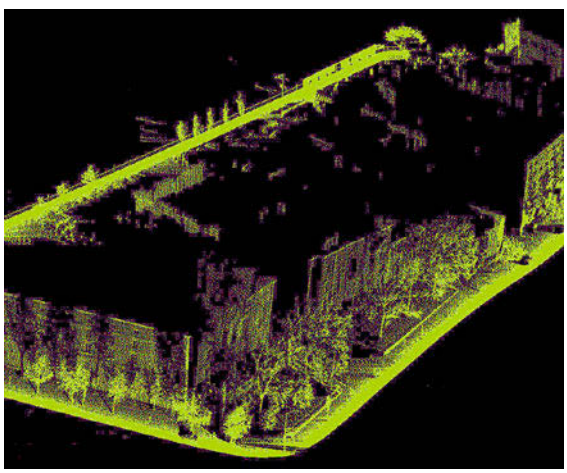


Fig. 45.22 Complex three-dimensional scene acquired with a mobile robot after filtering dynamic objects (after [45.74])

45.5 Summary and Further Reading

Further reading about typical representations and how they can be utilized for mobile robot navigation can be found in recent textbooks on mobile robotics [45.2, 79, 80]. A comprehensive survey of the fundamentals of spatial data structures and their applications is available in *Samet* [45.81].

Video-References

VIDEO 79 OctoMap visualization
available from http://handbookofrobotics.org/view-chapter/45/videodetails/79

VIDEO 269 3-D textured model of urban environments
available from http://handbookofrobotics.org/view-chapter/45/videodetails/269

VIDEO 270 Service robot navigation in urban environments
available from http://handbookofrobotics.org/view-chapter/45/videodetails/270

VIDEO 271 Learning navigation cost grids
available from http://handbookofrobotics.org/view-chapter/45/videodetails/271

References

45.1 H.P. Moravec, A.E. Elfes: High resolution maps from wide angle sonar, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1985)

45.2 H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun: *Principles of Robot Motion: Theory, Algorithms and Implementation* (MIT Press, Cambridge 2005)

45.3 D.H. Douglas, T.K. Peucker: Algorithms for the reduction of the number of points required to represent a line or its caricature, Cdn. Cartogr. **10**(2), 112–122 (1973)

45.4 D. Sack, W. Burgard: A comparison of methods for line extraction from range data, Proc. IFAC Symp. Intell. Auton. Veh. (IAV) (2004)

45.5 P. Beeson, N.K. Jong, B. Kuipers: Towards autonomous topological place detection using the extended Voronoi graph, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

45.6 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, Proc. Nat. Conf. Artif. Intell. (AAAI) (2002)

45.7 B.J. Kuipers, Y.-T. Byun: A robust qualitative method for spatial learning in unknown environments, Proc. Nat. Conf. Artif. Intell. (AAAI) (1988)

45.8 H. Choset, K. Nagatani: Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization, IEEE Trans. Robotics Autom. **17**(2), 125–137 (2001)

45.9 S. Thrun: Robotic mapping: A survey. In: *Exploring Artificial Intelligence in the New Millenium*, ed. by G. Lakemeyer, B. Nebel (Morgan Kaufmann, San Diego 2003)

45.10 M. Maimone, P. Leger, J. Biesiadecki: Overview of the Mars exploration rovers' autonomous mobility and vision capabilities, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)

45.11 S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, R. Chatila: Autonomous rover navigation on unknown terrains: functions and integration, Int. J. Robotics Res. **21**(10-11), 917–942 (2002)

45.12 R. Olea: *Geostatistics for Engineers and Earth Scientists* (Kluwer, Boston 1999)

45.13 A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarki, P. Rander, S. Thayer, N. Vallidi, R. Warner: Toward reliable off road autonomous vehicles operating in challenging environments, Int. J. Robotics Res. **25**(5-6), 449–483 (2006)

45.14 I.S. Kweon, T. Kanade: High-resolution terrain map from multiple sensor data, IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 278–292 (1992)

45.15 M. Montemerlo, S. Thrun: A multi-resolution pyramid for outdoor robot terrain perception, Proc. AAAI Nat. Conf. Artif. Intel., San Jose (2004)

45.16 R. Triebel, P. Pfaff, W. Burgard: Multi-level surface maps for outdoor terrain mapping and loop closing, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2006)

45.17 C. Wellington, A. Courville, A. Stentz: A generative model of terrain for autonomous navigation in vegetation, Int. J. Robotics Res. **25**(12), 1287–1304 (2006)

45.18 P. Pfaff, R. Triebel, W. Burgard: An efficient extension to elevation maps for outdoor terrain mapping and loop closing, Int. J. Robotics Res. **26**(2), 217–230 (2007)

45.19 D.M. Cole, P.M. Newman: Using laser range data for 3D SLAM in outdoor environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

45.20 A. Nüchter, K. Lingemann, J. Hertzberg, H. Surmann: 6D SLAM – 3D mapping outdoor environments: Research articles, J. Field Robotics **24**(8-9), 699–722 (2007)

45.21 J. Elseberg, D. Borrmann, A. Nüchter: Efficient processing of large 3D point clouds, Proc. 23rd Int. Symp. Infor. Commun. Autom. Technol. (ICAT) (2011)

45.22 R.B. Rusu, S. Cousins: 3D is here: Point cloud library (PCL), Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

45.23 N. Fairfield, G. Kantor, D. Wettergreen: Real-time SLAM with octree evidence grids for exploration in underwater tunnels, J. Field Robotics **24**(1), 3–21 (2007)

45.24 A. Foessel: Scene Modeling from Motion-Free Radar Sensing, Ph.D. Thesis (Carnegie Mellon Univ., Pittsburgh 2002)

45.25 K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, W. Burgard: OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems, Proc. ICRA Workshop Best Pract. 3D Percept. Model. Mob. Manip. (2010)

45.26 J.-F. Lalonde, N. Vandapel, M. Hebert: Data structure for efficient processing in 3-D, Proc. Robotics Sci. Syst., Vol. I (2005) p. 48

45.27 A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard: OctoMap: An efficient probabilistic 3D mapping framework based on octrees, Auton. Robots **34**(3), 189–206 (2013)

45.28 K.M. Wurm, D. Hennes, D. Holz, R.B. Rusu, C. Stachniss, K. Konolige, W. Burgard: Hierarchies of octrees for efficient 3D mapping, Proc. IEEE/RSJ Int Conf. Intell. Robots Syst. (IROS) (2011)

45.29 L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, M. Pollefeys: Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using onboard processing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

45.30 S. Oßwald, A. Hornung, M. Bennewitz: Improved proposals for highly accurate localization using range and vision data, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

45.31 M. Ciocarlie, K. Hsiao, E.G. Jones, S. Chitta, R.B. Rusu, I.A. Sucan: Towards reliable grasping and manipulation in household environments, Int. Symp. Exp. Robotics (ISER) (2010)

45.32 A. Hornung, M. Phillips, E.G. Jones, M. Bennewitz, M. Likhachev, S. Chitta: Navigation in three-dimensional cluttered environments for mobile manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

45.33 J. Leal: Stochastic Environment Representation, Ph.D. Thesis (Univ. of Sydney, Sydney 2003)

45.34 P. Heckbert, M. Garland: Optimal triangulation and quadric-based surface simplification, J. Comput. Geom. Theory Appl. **14**(1-3), 49–65 (1999)

45.35 A. Akbarzadeh: Towards urban 3d reconstruction from video, Proc. Int. Symp. 3D Data Vis. Transm. (2006)

45.36 C. Frueh, S. Jain, A. Zakhor: Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images, Int. J. Comput. Vis. **61**(2), 159–184 (2005)

45.37 I. Stamos, P. Allen: Geometry and texture recovery of scenes of large scales, Comput. Vis. Image Underst. **88**, 94–118 (2002)

45.38 D. Gennery: Traversability analysis and path planning for a planetary rover, Auton. Robotics **6**, 131–146 (1999)

45.39 D. Ferguson, A. Stentz: The delayed D* algorithm for efficient path replanning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

45.40 D. Ferguson, A. Stentz: Field D*: An interpolation-based path planner and replanner, Proc. Int. Symp. Robotics Res. (ISRR) (2005)

45.41 M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, S. Thrun: Anytime dynamic A*: An anytime, replanning algorithm, Proc. Int. Conf. Autom. Plan. Sched. (ICAPS) (2005)

45.42 B. Sofman, E. Lin, J. Bagnell, J. Cole, N. Vandapel, A. Stentz: Improving robot navigation through self-supervised online learning, J. Field Robotics **23**(12), 1059–1075 (2006)

45.43 D. Stavens, S. Thrun: A self-supervised terrain roughness estimator for off-road autonomous driving, Uncertainty Artif. Intell., Boston (2006)

45.44 A. Angelova, L. Matthies, D. Helmick, P. Perona: Slip prediction using visual information, Robotics Sci. Syst., Philadelphia (2006)

45.45 D. Kim, J. Sun, S. Oh, J. Rehg, A. Bobick: Traversability classification using unsupervised on-line visual learning for outdoor robot navigation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

45.46 S. Thrun, M. Montemerlo, A. Aron: Probabilistic terrain analysis for high-speed desert driving, Robotics Sci. Syst. (2005)

45.47 R. Murrieta-Cid, C. Parra, M. Devy: Visual navigation in natural environments: From range and color data to a landmark-based model, Auton. Robotics **13**(2), 143–168 (2002)

45.48 D. Asmar, J. Zelek, S. Abdallah: Tree trunks as landmarks for outdoor vision SLAM, Proc. Conf. Comp. Vis. Pattern Recogn. Workshop (2006)

45.49 I. Posner, D. Schroeter, P. Newman: Using scene similarity for place labelling, Int. Symp. Exp. Robotics (2006)

45.50 A. Torralba, K.P. Murphy, W.T. Freeman, M.A. Rubin: Context-based vision system for place and object recognition, Proc. IEEE Int. Conf. Comput. Vis. (ICCV) (2003)

45.51 D. Bradley, S. Thayer, A. Stentz, P. Rander: *Vegetation Detection for Mobile Robot Navigation*, *Tech. Rep. CMU-RI-TR-04-12* (Carnegie Mellon Univ., Pittsburgh 2004)

45.52 S. Kumar, J. Guivant, H. Durrant-Whyte: Informative representations of unstructured environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

45.53 S. Kumar, F. Ramos, B. Douillard, M. Ridley, H. Durrant-Whyte: A novel visual perception framework, Proc. 9th Int. Conf. Control Autom. Robotics Vis. (ICARCV) (2006)

45.54 F. Ramos, S. Kumar, B. Upcroft, H. Durrant-Whyte: Representing natural objects in unstructured environments, NIPS Workshop Mach. Learn. Robotics (2005)

45.55 C. Pantofaru, R. Unnikrishnan, M. Hebert: Toward generating labeled maps from color and range data for robot navigation, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (2003)

45.56 D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, A. Ng: Discriminative learning of Markov random fields for segmentation of 3D scan data, Proc. Conf. Comp. Vis. Pattern Recogn. (CVPR) (2005)

45.57 R. Triebel, K. Kersting, W. Burgard: Robust 3D scan point classification using associative Markov networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

45.58 J.F. Lalonde, N. Vandapel, D. Huber, M. Hebert: Natural terrain classification using three-dimensional ladar data for ground robot mobility, J. Field Robotics **23**(10), 839–861 (2006)

45.59 M. Devy, R. Chatila, P. Fillatreau, S. Lacroix, F. Nashashibi: On autonomous navigation in a natural environment, Robotics Auton. Syst. **16**(1), 5–16 (1995)

Avidbots Ex1016
Page 1179 of 2154

45.60 R. Manduchi, A. Castano, A. Talukder, L. Matthies: Obstacle detection and terrain classification for autonomous off-road navigation, Auton. Robotics **18**(1), 81–102 (2005)

45.61 D. Huber, M. Hebert: 3D modeling using a statistical sensor model and stochastic search, Proc. IEEE Conf. Comput. Vision Pattern Recogn. (CVPR) (2003) pp. 858–865

45.62 S. Balakirsky, A. Lacaze: World modeling and behavior generation for autonomous ground vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000)

45.63 A. Lacaze, K. Murphy, M. Delgiorno: Autonomous mobility for the demo III experimental unmanned vehicles, Proc. AUVSI Int. Conf. Unmanned Veh. (2002)

45.64 P. Bellutta, R. Manduchi, L. Matthies, K. Owens, A. Rankin: Terrain perception for demo III, Proc. Intell. Veh. Symp. (2000)

45.65 J.F. Lalonde, R. Unnikrishnan, N. Vandapel, M. Hebert: Scale selection for classification of point-sampled 3D surfaces, Proc. 5th Int. Conf. 3-D Digital Imaging Model. (3DIM) (2005)

45.66 J. Macedo, R. Manduchi, L. Matthies: Ladar-based discrimination of grass from obstacles for autonomous navigation, Proc. 7th Int. Symp. Exp. Robotics (ISER) (2000)

45.67 H. Chen, P. Meer, D. Tyler: Robust regression for data with multiple structures, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn. (CVPR) (2001)

45.68 R. Unnikrishnan, M. Hebert: Robust extraction of multiple structures from non-uniformly sampled data, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2003)

45.69 D. Wolf, G. Sukhatme, D. Fox, W. Burgard: Autonomous terrain mapping and classification using hidden Markov models, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

45.70 C. Olson, L. Matthies, J. Wright, R. Li, K. Di: Visual terrain mapping for Mars exploration, Comput. Vis. Underst. **105**, 73–85 (2007)

45.71 J. Nieto, J. Guivant, E. Nebot: The hybrid metric maps (hymms): A novel map representation for denseSLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

45.72 F. Ramos, J. Nieto, H. Durrant-Whyte: Recognising and modelling landmarks to close loops in outdoor SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)

45.73 D. Hähnel, D. Schulz, W. Burgard: Mobile robot mapping in populated environments, Adv. Robotics **17**(7), 579–598 (2003)

45.74 C.-C. Wang, C. Thorpe, S. Thrun: Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

45.75 P. Biber, T. Duckett: Dynamic maps for long-term operation of mobile service robots, Proc. Robotics Sci. Syst. (2005)

45.76 D. Meyer-Delius, J. Hess, G. Grisetti, W. Burgard: Temporary maps for robust localization in semi-static environments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Taipei (2010)

45.77 C. Stachniss, W. Burgard: Mobile robot mapping and localization in non-static environments, Proc. Nat. Conf. Artif. Intell. (AAAI), Pittsburgh (2005)

45.78 D. Hähnel, R. Triebel, W. Burgard, S. Thrun: Map building with mobile robots in dynamic environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

45.79 R. Siegwart, I. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2001)

45.80 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

45.81 H. Samet: *Foundations of Multidimensional and Metric Data Structures* (Elsevier, Amsterdam 2006)

Avidbots Ex1016
Page 1180 of 2154

Multimedia Contents

46. Simultaneous Localization and Mapping

Cyrill Stachniss, John J. Leonard, Sebastian Thrun

This chapter provides a comprehensive introduction in to the *simultaneous localization and mapping problem*, better known in its abbreviated form as SLAM. SLAM addresses the main perception problem of a robot navigating an unknown environment. While navigating the environment, the robot seeks to acquire a map thereof, and at the same time it wishes to localize itself using its map. The use of SLAM problems can be motivated in two different ways: one might be interested in detailed environment models, or one might seek to maintain an accurate sense of a mobile robot's location. SLAM serves both of these purposes.

We review the three major paradigms from which many published methods for SLAM are derived: (1) the extended Kalman filter (EKF); (2) particle filtering; and (3) graph optimization. We also review recent work in three-dimensional (3-D) SLAM using visual and red green blue distance-sensors (RGB-D), and close with a discussion of open research problems in robotic mapping.

46.1 **SLAM: Problem Definition** 1154
46.1.1 Mathematical Basis 1154
46.1.2 Example: SLAM in Landmark Worlds 1155
46.1.3 Taxonomy of the SLAM Problem .. 1156
46.2 **The Three Main SLAM Paradigms** 1157
46.2.1 Extended Kalman Filters 1157
46.2.2 Particle Methods 1159
46.2.3 Graph-Based Optimization Techniques..... 1162
46.2.4 Relation of Paradigms..... 1166
46.3 **Visual and RGB-D SLAM** 1166
46.4 **Conclusion and Future Challenges** 1169
Video-References..... 1170
References..... 1171

This chapter provides a comprehensive introduction into one of the key enabling technologies of mobile robot navigation: *simultaneous localization and mapping*, or in short *SLAM*. SLAM addresses the problem of acquiring a spatial map of an environment while simultaneously localizing the robot relative to this model. The SLAM problem is generally regarded as one of the most important problems in the pursuit of building truly autonomous mobile robots. It is of great practical importance; if a robust, general-purpose solution to SLAM can be found, then many new applications of mobile robotics will become possible.

While the problem is deceptively easy to state, it presents many challenges, despite significant progress made in this area. At present, we have robust methods for mapping environments that are mainly static, structured, and of limited size. Robustly mapping unstructured, dynamic, and large-scale environments in an online fashion remains largely an open research problem.

The historical roots of methods that can be applied to address the SLAM problem can be traced back to Gauss [46.1], who is largely credited for inventing the least squares method. In the Twentieth Century, a number of fields outside robotics have studied the making of environment models from a moving sensor platform, most notably in *photogrammetry* [46.2–4] and *computer vision* [46.5]. Strongly related problems in these fields are bundle adjustment and structure from motion. SLAM builds on this work, often extending the basic paradigms into more scalable algorithms. Modern SLAM systems often view the estimation problem as solving a sparse graph of constraints and applying

nonlinear optimization to compute the map and the trajectory of the robot. As we strive to enable long-lived autonomous robots, an emerging challenge is to handle massive sensor data streams.

This chapter begins with a definition of the SLAM problem, which shall include a brief taxonomy of different versions of the problem. The centerpiece of this chapter is a layman introduction into the three major paradigms in this field, and the various extensions that exist. As the reader will quickly recognize, there is no single best solution to the SLAM method. The method chosen by the practitioner will depend on a number of factors, such as the desired map resolution, the update time, and the nature of the features in the map, and so on. Nevertheless, the three methods discussed in this chapter cover the major paradigms in this field.

For more a detailed treatment of SLAM, we refer the reader to Durrant-Whyte and Bailey [46.6, 7], who provide an in-depth tutorial for SLAM, Grisetti et al. for a tutorial on graph-based SLAM [46.8], and Thrun et al., which dedicates a number of chapters to the topic of SLAM [46.9].

46.1 SLAM: Problem Definition

The SLAM problem is defined as follows: A mobile robot roams an unknown environment, starting at an initial location x_0. Its motion is uncertain, making it gradually more difficult to determine its current pose in global coordinates. As it roams, the robot can sense its environment with a noisy sensor. The SLAM problem is the problem of building a map of the environment while simultaneously determining the robot's position relative to this map given noisy data.

46.1.1 Mathematical Basis

Formally, SLAM is best described in probabilistic terminology. Let us denote time by t, and the robot location by x_t. For mobile robots on a flat ground, x_t is usually a three-dimensional vector, comprising its two-dimensional (2-D) coordinate in the plane plus a single rotational value for its orientation. The sequence of locations, or *path*, is then given as

$$X_T = \{x_0, x_1, x_2, \ldots, x_T\} \,. \tag{46.1}$$

Here T is some terminal time (T might be ∞). The initial location x_0 often serves as a point of reference for the estimation algorithm; other positions cannot be sensed.

Odometry provides relative information between two consecutive locations. Let u_t denote the odometry that characterized the motion between time $t-1$ and time t; such data might be obtained from the robot's wheel encoders or from the controls given to those motors. Then the sequence

$$U_T = \{u_1, u_2, u_3 \ldots, u_T\} \tag{46.2}$$

characterizes the relative motion of the robot. For noise-free motion, U_T would be sufficient to recover the poses from the initial location x_0. However, odometry measurements are noisy, and path integration techniques inevitably diverge from the truth.

Finally, the robot senses objects in the environment. Let m denote the *true* map of the environment. The environment may be comprised of landmarks, objects, surfaces, etc., and m describes their locations. The environment map m is often assumed to be time-invariant, i. e., static.

The robot measurements establish information between features in m and the robot location x_t. If we, without loss of generality, assume that the robot takes exactly one measurement at each point in time, the sequence of measurements is given as

$$Z_T = \{z_1, z_2, z_3, \ldots, z_T\} \,. \tag{46.3}$$

Figure 46.1 illustrates the variables involved in the SLAM problem. It shows the sequence of locations and sensor measurements, and the causal relationships between these variables. This diagram represents a *graph-*

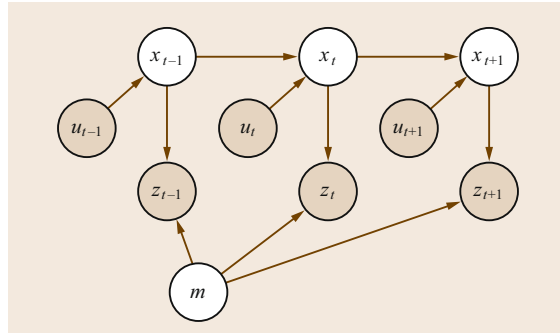


Fig. 46.1 Graphical model of the SLAM problem. Arcs indicate causal relationships, and shaded nodes are directly observable to the robot. In SLAM, the robot seeks to recover the unobservable variables

ical model for SLAM. It is useful in understanding the dependencies in the problem at hand.

The SLAM problem is now the problem of recovering a model of the world m and the sequence of robot locations X_T from the odometry and measurement data. The literature distinguishes two main forms of the SLAM problem, which are both of equal practical importance. One is known as the *full SLAM problem*: it involves estimating the posterior over the entire robot path together with the map

$$p(X_T, m \mid Z_T, U_T) \,. \tag{46.4}$$

Written in this way, the full SLAM problem is the problem of calculating the joint posterior probability over X_T and m from the available data. Notice that the variables right of the conditioning bar are all directly observable to the robot, whereas those on the left are the ones that we want. As we shall see, algorithms for the full SLAM problem are often batch, that is, they process all data at the same time.

The second, equally important SLAM problem is the *online SLAM problem*. This problem is defined via

$$p(x_t, m \mid Z_t, U_t) \,. \tag{46.5}$$

Online SLAM seeks to recover the present robot location, instead of the entire path. Algorithms that address the online problem are usually incremental and can process one data item at a time. In the literature, such algorithms are typically called *filters*.

To *solve* the SLAM problem, the robot needs to be endowed with two more models: a mathematical model that relates odometry measurements u_t to robot locations x_{t-1} and x_t; and a model that relates measurements z_t to the environment m and the robot location x_t. These models correspond to the arcs in Fig. 46.1.

In SLAM, it is common to think of those mathematical models as probability distributions: $p(x_t \mid x_{t-1}, u_t)$ characterizes the probability distribution of the location x_t assuming that a robot started at a known location x_{t-1} and measured the odometry data u_t. And likewise, $p(z_t \mid x_t, m)$ is the probability for measuring z_t if this measurement is taken at a known location x_t in a known environment m. Of course, in the SLAM problem we do *not* know the robot location, and neither do we know the environment. As we shall see, Bayes rule takes care of this, by transforming these mathematical relationships into a form where we can *recover* probability distributions over those latent variables from the measured data.

46.1.2 Example: SLAM in Landmark Worlds

One common setting of SLAM involves an assumption that the environment is populated by point-landmarks. When building 2-D maps, point-landmarks may correspond to door posts and corners of rooms, which, when projected into a 2-D map are characterized by a point coordinate. In a 2-D world, each point-landmark is characterized by two coordinate values. Hence the world is a vector of size $2N$, where N is the number of point-landmarks in the world. In a commonly studied setting, the robot can sense three things: the relative range to nearby landmarks, their relative bearing, and the identity of these landmarks. The range and bearing may be noisy, but in the most simple case the identity of the sensed landmarks is known perfectly. Determining the identity of the sensed landmarks is also known as the *data association problem*. In practice, it is one of the most difficult problems in SLAM.

To model the above described setup, one begins with defining the *exact*, noise-free measurement function. The measurement function h describes the workings of the sensors: it accepts as input a description of the environment m and a robot location x_t, and it computes the measurement

$$h(x_t, m) \,. \tag{46.6}$$

Computing h is straightforward in our simplified landmark setting; it is a simple exercise in trigonometry. The probabilistic measurement model can be derived from this measurement function by adding a noise term. It is a probability distribution that peaks at the noise-free value $h(x_t, m)$ but allows for measurement noise, for example,

$$p(z_t \mid x_t, m) = \mathcal{N}(h(x_t, m), \mathbf{Q}_t) \,. \tag{46.7}$$

Here $\mathcal{N}$ denotes the 2-D normal distribution, which is centered at $h(x_t, m)$. The 2-by-2 matrix $\mathbf{Q}_t$ is the noise covariance, indexed by time.

The motion model is derived from a kinematic model of robot motion. Given the location vector x_{t-1} and the motion u_t, textbook kinematics tells us how to calculate x_t. Let this function be denoted by g

$$g(x_{t-1}, u_t) \,. \tag{46.8}$$

The motion model may then be defined by a normal distribution centered at $g(x_{t-1}, u_t)$ but subject to Gaussian noise

$$p(x_t \mid x_{t-1}, u_t) = \mathcal{N}(g(x_{t-1}, u_t), \mathbf{R}_t) \,. \tag{46.9}$$

Here $\mathbf{R}_t$ is a covariance. It is of size 3-by-3, since the location is a three-dimensional 3-D vector.

With these definitions, we have all we need to develop a SLAM algorithm. While in the literature,

point-landmark problems with range-bearing sensing are by far the most studied, SLAM algorithms are not confined to landmark worlds. But no matter what the map representation and the sensor modality, any SLAM algorithm needs a similarly crisp definition of the features in m, the measurement model $p(z_t \mid x_t, m)$, and the motion model $p(x_t \mid x_{t-1}, u_t)$. Note that none of those distributions has to be restricted to Gaussian noise as done in the example above.

46.1.3 Taxonomy of the SLAM Problem

SLAM problems are distinguished along a number of different dimensions. Most important research papers identify the type of problems addressed by making the underlying assumptions explicit. We already encountered one such distinction: full versus online. Other common distinctions are as follows:

Volumetric Versus Feature-Based

In volumetric SLAM, the map is sampled at a resolution high enough to allow for photo-realistic reconstruction of the environment. The map m in volumetric SLAM is usually quite high-dimensional, with the result that the computation can be quite involved. Feature-based SLAM extracts sparse features from the sensor stream. The map is then only comprised of features. Our point-landmark example is an instance of feature-based SLAM. Feature-based SLAM techniques tend to be more efficient, but their results may be inferior to volumetric SLAM due to the fact that the extraction of features discards information in the sensor measurements.

Topological Versus Metric

Some mapping techniques recover only a qualitative description of the environment, which characterizes the relation of basic locations. Such methods are known as topological. A topological map might be defined over a set of distinct places and a set of qualitative relations between these places (e.g., place A is adjacent to place B). Metric SLAM methods provide metric information between the relation of such places. In recent years, topological methods have fallen out of fashion, despite ample evidence that humans often use topological information for navigation.

Known Versus Unknown Correspondence

The correspondence problem is the problem of relating the identity of sensed things to other sensed things. In the landmark example above, we assumed that the identity of landmarks is known. Some SLAM algorithms make such an assumption, others do not. The ones that do not provide special mechanisms for estimating the correspondence of measured features to previously observed landmarks in the map. The problem of estimating the correspondence is known as *data association problem*. It is one of the most difficult problems in SLAM.

Static Versus Dynamic

Static SLAM algorithms assume that the environment does not change over time. Dynamic methods allow for changes in the environment. The vast literature on SLAM assumes static environments. Dynamic effects are often treated just as measurement outliers. Methods that reason about motion in the environment are more involved, but they tend to be more robust in most applications.

Small Versus Large Uncertainty

SLAM problems are distinguished by the degree of location uncertainty that they can handle. The most simple SLAM algorithms allow only for small errors in the location estimate. They are good for situations in which a robot goes down a path that does not intersect itself, and then returns along the same path. In many environments it is possible to reach the same location from multiple directions. Here the robot may accrue a large amount of uncertainty. This problem is known as the *loop closing problem*. When closing a loop, the uncertainty may be large. The ability to close loops is a key characteristic of modern-day SLAM algorithms. The uncertainty can be reduced if the robot can sense information about its position in some absolute coordinate frame, e.g., through the use of a satellite-based global positioning system (GPS) receiver.

Active Versus Passive

In passive SLAM algorithms, some other entity controls the robot, and the SLAM algorithm is purely observing. The vast majority of algorithms are of this type; they give the robot designer the freedom to implement arbitrary motion controllers, and pursue arbitrary motion objectives. In active SLAM, the robot actively explores its environment in the pursuit of an accurate map. Active SLAM methods tend to yield more accurate maps in less time, but they constrain the robot motion. There exist hybrid techniques in which the SLAM algorithm controls only the pointing direction of the robot's sensors, but not the motion direction.

Single-Robot Versus Multi-Robot

Most SLAM problems are defined for a single robot platform, although recently the problem of multi-robot exploration has gained in popularity. Multi-robot SLAM problems come in many flavors. In some, robots get to observe each other, in others, robots are told their relative initial locations. Multirobot SLAM problems

are also distinguished by the type of communication allowed between the different robots. In some, the robots can communicate with no latency and infinite bandwidth. More realistic are setups in which only nearby robots can communicate, and the communication is subject to latency and bandwidth limitations.

Any-Time and Any-Space

Robots that do all computations onboard have limited resources in memory and computation power. Anytime and any-space SLAM systems are an alternative to traditional methods. They enable the robot to compute a solution given the resource constraints of the system. The more resources available, the better the solution.

As this taxonomy suggests, there exists a flurry of SLAM algorithms. Most modern-day conferences dedicate multiple sessions to SLAM. This chapter focuses on the very basic SLAM setup. In particular it assumes a static environment with a single robot. Extensions are discussed towards the end of this chapter, in which the relevant literature is discussed.

46.2 The Three Main SLAM Paradigms

This section reviews three basic SLAM paradigms, from which most others are derived. The first, known as EKF SLAM, is in robotics historically the earliest but has become less popular due to its limiting computational properties and issues resulting from performing single linearizations only. The second approach uses nonparametric statistical filtering techniques known as particle filters. It is a popular method for online SLAM and provides a perspective on addressing the data association problem in SLAM. The third paradigm is based on graphical representations and successfully applies sparse nonlinear optimization methods to the SLAM problem. It is the main paradigm for solving the full SLAM problem and recently also incremental techniques are available.

46.2.1 Extended Kalman Filters

Historically, the EKF formulation of SLAM is the earliest, and perhaps the most influential, SLAM algorithm. EKF SLAM was introduced in [46.10, 11] and [46.12, 13], which were the first papers to propose the use of a single state vector to estimate the locations of the robot and a set of features in the environment, with an associated error covariance matrix representing the uncertainty in these estimates, including the correlations between the vehicle and feature state estimates. As the robot moves through its environment taking measurements, the system state vector and covariance matrix are updated using the extended Kalman filter [46.14, 15]. As new features are observed, new states are added to the system state vector; the size of the system covariance matrix grows quadratically.

This approach assumes a metrical, feature-based environmental representation, in which objects can be effectively represented as points in an appropriate parameter space. The position of the robot and the locations of features form a network of uncertain spatial relationships. The development of appropriate representations is a critical issue in SLAM, and intimately related to the topics of sensing and world modeling discussed in Chap. 36 and in Part C.

The EKF algorithm represents the robot estimate by a multivariate Gaussian

$$p(x_t, m \mid Z_t, U_t) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \,. \tag{46.10}$$

The high-dimensional vector $\boldsymbol{\mu}_t$ contains the robot's best estimate of its own current location x_t and the location of the features in the environment. In our point-landmark example, the dimension of $\boldsymbol{\mu}_t$ would be $3 + 2N$, since we need three variables to represent the robot location and $2N$ variables for the N landmarks in the map.

The matrix $\boldsymbol{\Sigma}_t$ is the covariance of the robot's assessment of its expected error in the guess $\boldsymbol{\mu}_t$. The matrix $\boldsymbol{\Sigma}_t$ is of size $(3 + 2N) \times (3 + 2N)$ and it is positive semi-definite. In SLAM, this matrix is usually dense. The off-diagonal elements capture the correlations in the estimates of different variables. Nonzero correlations come along because the robot's location is uncertain, and as a result the locations of the landmarks in the maps are uncertain.

The EKF SLAM algorithm is easily derived for our point-landmark example. Suppose, for a moment, the motion function g and the measurement function h were *linear* in their arguments. Then, the vanilla Kalman filter, as described in any textbook on Kalman filtering, would be applicable. EKF SLAM linearizes the functions g and h using Taylor series expansion. In its most basic form and in the absence of any data association problems, EKF SLAM is basically the application of the EKF to the online SLAM problem.

Figure 46.2 illustrates the EKF SLAM algorithm for an artificial example. The robot navigates from a start pose that serves as the origin of its coordinate system. As it moves, its own pose uncertainty increases, as indicated by uncertainty ellipses of growing diameter. It

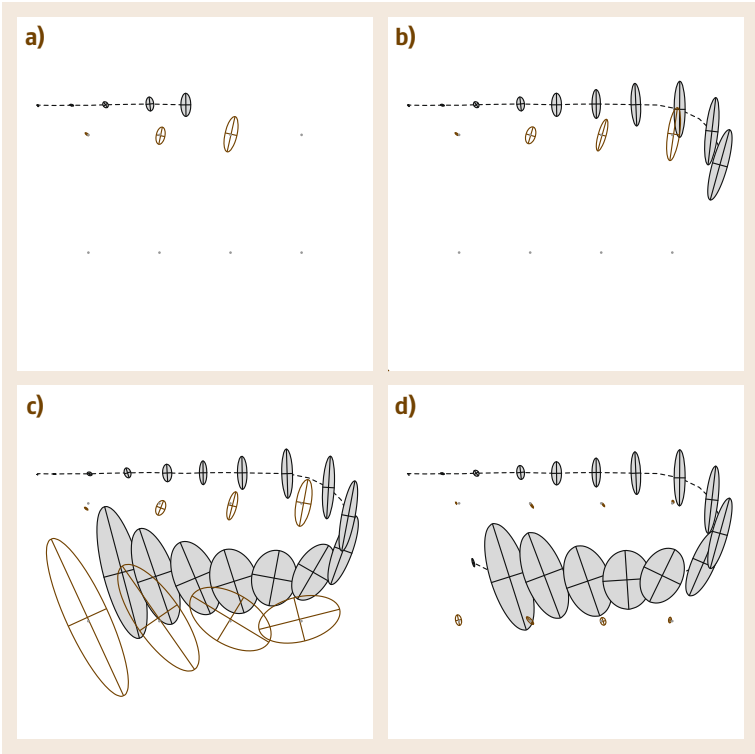


Fig.46.2a–d EKF applied to the online SLAM problem. The robot's path is a *dotted line*, and its estimates of its own position are *shaded ellipses*. Eight distinguishable landmarks of unknown location are shown as *small dots*, and their location estimates are shown as *white ellipses*. In **(a–c)** the robot's positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In **(d)** the robot senses the first landmark again, and the uncertainty of *all* landmarks decreases, as does the uncertainty of its current pose (image courtesy of Michael Montemerlo, Stanford University)

also senses nearby landmarks and maps them with an uncertainty that combines the fixed measurement uncertainty with the increasing pose uncertainty. As a result, the uncertainty in the landmark locations grows over time. The interesting transition happens in Fig. 46.2d: Here the robot observes the landmark it saw in the very beginning of mapping, and whose location is relatively well known. Through this observation, the robot's pose error is reduced, as indicated in Fig. 46.2d – notice the very small error ellipse for the final robot pose. This observation also reduces the uncertainty for other landmarks in the map. This phenomenon arises from a correlation that is expressed in the covariance matrix of the Gaussian posterior. Since most of the uncertainty in earlier landmark estimates is caused by the robot pose, and since this very uncertainty persists over time, the location estimates of those landmarks are correlated. When gaining information on the robot's pose, this information spreads to previously observed landmarks. This effect is probably the most important characteristic of the SLAM posterior [46.16]. Information that helps localize the robot is propagated through the map, and as a result improves the localization of other landmarks in the map.

With a few adaptations, EKF SLAM can also be applied in the presence of *uncertain data association*. If the identity of observed features is unknown, the basic EKF idea becomes inapplicable. The solution here is to reason about the most likely data association when a landmark is observed. This is usually done based on proximity: *which of the landmarks in the map corresponds most likely to the landmark just observed?* The proximity calculation considers the measurement noise and the actual uncertainty in the poster estimate, and the metric used in this calculation is known as a Mahalanobis distance, which is a weighted quadratic distance. To minimize the chances of false data associations, many implementations use visible features to distinguish individual landmarks and associate groups of landmarks observed simultaneously [46.17, 18], although distinct features can also be computed from laser data [46.19, 20]. Typical implementations also maintain a *provisional landmark list* and only add landmarks to the internal map when they have been observed sufficiently frequently [46.16, 21]. With an appropriate landmark definition and careful implementation of the data association step, EKF SLAM has been applied successfully in a wide range of environments, using airborne, underwater, indoor, and various other platforms.

The basic formulation of EKF SLAM assumes that the location of features in the map is fully observable from a single position of the robot. The method has been extended to situations with partial observability,

with range-only [46.22] or angle-only [46.23, 24] measurements. The technique has also been utilized using a feature-less representation, in which the state consists of current and past robot poses, and measurements take the form of constraints between the poses (derived for example from laser scan matching or from camera measurements) [46.25, 26].

A key concern of the EKF approach to SLAM lies in the quadratic nature of the covariance matrix. A number of researchers have proposed extensions to the EKF SLAM algorithms that achieve scalability, for example through submap decomposition [46.27–30]. A related family of approaches [46.31–34] employs the Extended Information Filter, which operates on the inverse of the covariance matrix. A key insight is that whereas the EKF covariance is densely populated, the information matrix is sparse when the full robot trajectory is maintained, leading to the development of efficient algorithms and providing a conceptual link to the pose graph optimization methods described in Sect. 46.2.3.

The issues of consistency and convergence in EKF SLAM have been investigated in [46.35, 36]. Observability-based rules for designing consistent EKF SLAM estimators are presented in [46.37].

46.2.2 Particle Methods

The second principal SLAM paradigm is based on particle filters. Particle filters can be traced back to [46.38], but they have become popular only in the last two decades. Particle filters represent a posterior through a set of *particles*. For the novice in SLAM, each particle is best thought as a concrete guess as to what the true value of the state may be. By collecting many such guesses into a set of guesses, or set of particles, the particle filter approximates the posterior distribution. Under mild conditions, the particle filter has been shown to approach the true posterior as the particle set size goes to infinity. It is also a nonparametric representation that represents multimodal distributions with ease.

The key problem with the particle filter in the context of SLAM is that the space of maps and robot paths is huge. Suppose we have a map with 100 features. How many particles would it take to populate that space? In fact, particle filters scale exponentially with the dimension of the underlying state space. Three or four dimensions are thus acceptable, but 100 dimensions are generally not.

The trick to make particle filters amenable to the SLAM problem goes back to [46.39, 40] and is known as Rao–Blackwellization. It has been introduced into the SLAM literature in [46.41], followed by [46.42], who coined the name fastSLAM (fast simultaneous localization and mapping). Let us first explain the basic FastSLAM algorithm on the simplified point-landmark example, and then discuss the justification for this approach.

At any point in time, FastSLAM maintains K particles of the type

$$X_t^{[k]}, \boldsymbol{\mu}_{t,1}^{[k]}, \ldots, \boldsymbol{\mu}_{t,N}^{[k]}, \boldsymbol{\Sigma}_{t,1}^{[k]}, \ldots, \boldsymbol{\Sigma}_{t,N}^{[k]}. \tag{46.11}$$

Here $[k]$ is the index of the sample. This expression states that a particle contains:

- A sample path $X_t^{[k]}$, and
- A set of N 2-D Gaussians with means $\boldsymbol{\mu}_{t,n}^{[k]}$ and variances $\boldsymbol{\Sigma}_{t,n}^{[k]}$), one for each landmark in the environment.

Here n is the index of the landmark (with $1 \leq n \leq N$). From that it follows that K particles possess K path samples. It also possesses KN Gaussians, each of which models exactly one landmark for one of the particles.

Initializing FastSLAM is simple: just set each particle's robot location to the starting coordinates, typically $(0, 0, 0)^{\mathrm{T}}$, and zero the map. The particle update then proceeds as follows:

- When an odometry reading is received, new location variables are generated stochastically, one for each of the particles. The distribution for generating those location particles is based on the motion model

$$x_t^{[k]} \approx p(x_t \mid x_{t-1}^{[k]}, u_t) \,. \tag{46.12}$$

  Here $x_{t-1}^{[k]}$ is the previous location, which is part of the particle. This probabilistic sampling step is easily implemented for any robot whose kinematics can be computed.
- When a measurement z_t is received, two things happen: first, FastSLAM computes for each particle the probability of the new measurement z_t. Let the index of the sensed landmark be n. Then the desired probability is defined as follows

$$w_t^{[k]} = \mathcal{N}(z_t \mid x_t^{[k]}, \boldsymbol{\mu}_{t,n}^{[k]}, \boldsymbol{\Sigma}_{t,n}^{[k]}) \,. \tag{46.13}$$

  The factor $w_t^{[k]}$ is called the *importance weight*, since it measures how *important* the particle is in the light of the new sensor measurement. As before, $\mathcal{N}$ denotes the normal distribution, but this time it is calculated for a specific value, z_t. The importance weights of all particles are then normalized so that they sum to 1.

  Next, FastSLAM draws with replacement from the set of existing particles a set of new particles. The probability of drawing a particle is its normalized

importance weight. This step is called *resampling*. The intuition behind resampling is that particles for which the measurement is more plausible have a higher chance of surviving the resampling process.

Finally, FastSLAM updates for the new particle set the mean $\boldsymbol{\mu}_{t,n}^{[k]}$ and covariance $\boldsymbol{\Sigma}_{t,n}^{[k]}$, based on the measurement z_t. This update follows the standard EKF update rules – note that the extended Kalman filters maintained in FastSLAM are, in contrast to EKF SLAM, all low-dimensional (typically 2-D).

This all may sound complex, but FastSLAM is quite easy to implement. Sampling from the motion model usually involves simple kinematic calculations. Computing the importance of a measurement is often straightforward too, especially for Gaussian measurement noise. And updating a low-dimensional particle filter is also not complicated.

FastSLAM has been shown to approximate the full SLAM posterior. The derivation of FastSLAM exploits three techniques: Rao–Blackwellization, conditional independence, and resampling. Rao–Blackwellization is the following concept. Suppose we would like to compute a probability distribution $p(a, b)$, where a and b are arbitrary random variables. The vanilla particle filter would draw particles from the joint distributions, that is, each particle would have a value for a and one for b. However, if the conditional $p(b \mid a)$ can be described in closed form, it is equally legitimate to just draw particles from $p(a)$, and attach to each particle a closed-form description of $p(b \mid a)$. This trick is known as Rao–Blackwellization, and it yields better results than sampling from the joint. FastSLAM applies this technique, in that it samples from the path posterior $p(X_t^{[k]} \mid U_t, Z_t)$ and represents the map $p(m \mid X_t^{[k]}, U_t, Z_t)$ in Gaussian form.

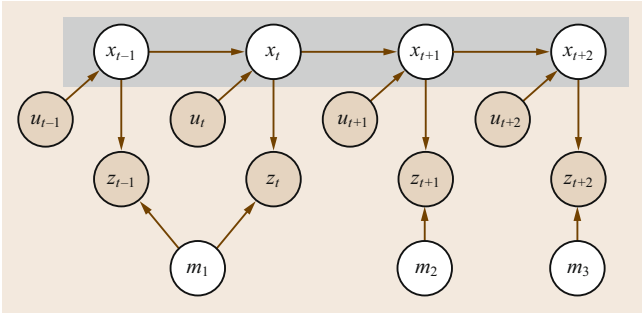


Fig. 46.3 The SLAM problem depicted as Bayes network graph. The robot moves from location x_{t-1} to location x_{t+2}, driven by a sequence of controls. At each location x_t it observes a nearby feature in the map $m = \{m_1, m_2, m_3\}$. This graphical network illustrates that the location variables *separate* the individual features in the map from each other. If the locations are known, there remains no other path involving variables whose value is not known, between any two features in the map. This lack of a path renders the posterior of any two features in the map conditionally independent (given the locations)

FastSLAM also breaks down the posterior over maps (conditioned on paths) into sequences of low-dimensional Gaussians. The justification for this decomposition is subtle. It arises from a specific conditional independence assumption that is native to SLAM. Fig. 46.3 illustrates the concept graphically. In SLAM, knowledge of the robot path renders all landmark estimates independent. This is easily shown for the graphical network in Fig. 46.3: we find that if we remove the path variables from Fig. 46.3, then the landmark variables are all disconnected [46.43]. Thus, in SLAM *any dependence between multiple landmark estimates is mediated through the robot path.* This subtle but important observation implies that even if we used a large, monolithic Gaussian for the entire map (one per particle, of course), the off-diagonal element between different landmarks would simply remain zero. It is therefore legitimate to implement the map more efficiently, using N small Gaussians, one for each landmark. This explains the efficient map representation in FastSLAM.

Figure 46.4 shows results for a point-feature problem; here the point features are the centers of tree trunks as observed by an outdoor robot. The dataset used here is known as the Victoria Park dataset [46.44]. Fig. 46.4a shows the path of the vehicle obtained by integrating the vehicle controls, without perception. As can be seen, controls are a poor predictor of location for this vehicle; after 30 min of driving, the estimated position of the vehicle is well over 100 m away from its GPS position.

The FastSLAM algorithm has a number of interesting properties. First, it solves both full and online SLAM problems. Each particle has a sample of an entire path but the actual update equation only uses the most recent pose. This makes FastSLAM a filter. Second, FastSLAM can maintain multiple data association hypotheses. It is straightforward to make data association decisions on a per-particle basis, instead of having to adopt the same hypothesis for the entire filter. While we will not give any mathematical justification, we note that the resulting FastSLAM algorithm can even deal with unknown data association – something that the extended Kalman filter cannot claim. And third, FastSLAM can be implemented very efficiently using advanced tree methods to represent the map estimates, the update can be performed in time logarithmic in the size of the map N, and linear in the number of particles M.

Avidbots Ex1016
Page 1188 of 2154

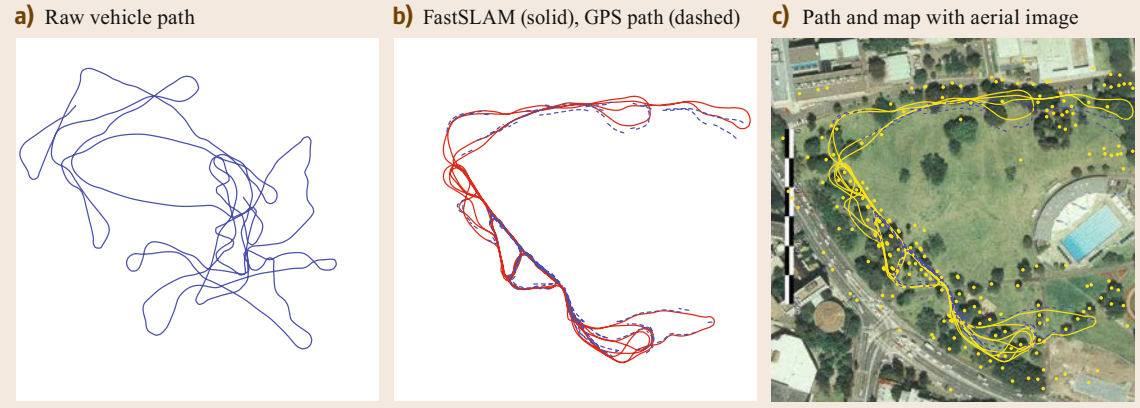


Fig. 46.4 (a) Vehicle path predicted by the odometry; (b) True path (*dashed line*) and FastSLAM 1.0 path (*solid line*); (c) Victoria Park results overlaid on aerial imagery with the GPS path in *blue* (*dashed*), average FastSLAM 1.0 path in *yellow* (*solid*), and estimated features as *yellow dots* (data and aerial image courtesy of José Guivant and Eduardo Nebot, Australian Centre for Field Robotics)



Fig. 46.5 Occupancy grid map generated from laser range data and based on pure odometry (image courtesy of Dirk Hähnel, University of Freiburg)

FastSLAM has been extended in several ways. One set of variants are grid-based versions of FastSLAM, in which the Gaussians used to model point landmarks are replaced by an occupancy grid map [46.45–47]. The variant of [46.46] is illustrated in Fig. 46.5.

Figure 46.6 illustrates a simplified situation with three particles just before closing a large loop. The three different particles each stand for different paths, and they also posses their own local maps. When the loop is closed importance resampling selects those particles whose maps are most consistent with the measurement. A resulting large-scale map is shown in Fig. 46.5. Further extensions can be found in [46.48, 49], whose methods are called DP-SLAM and operate on ancestry trees to provide efficient tree update methods for grid-based maps. Related to that, approximations to FastSLAM in which particles share their maps have been proposed [46.50].

The works in [46.45, 47, 51] provide ways to incorporate new observations into the location sampling

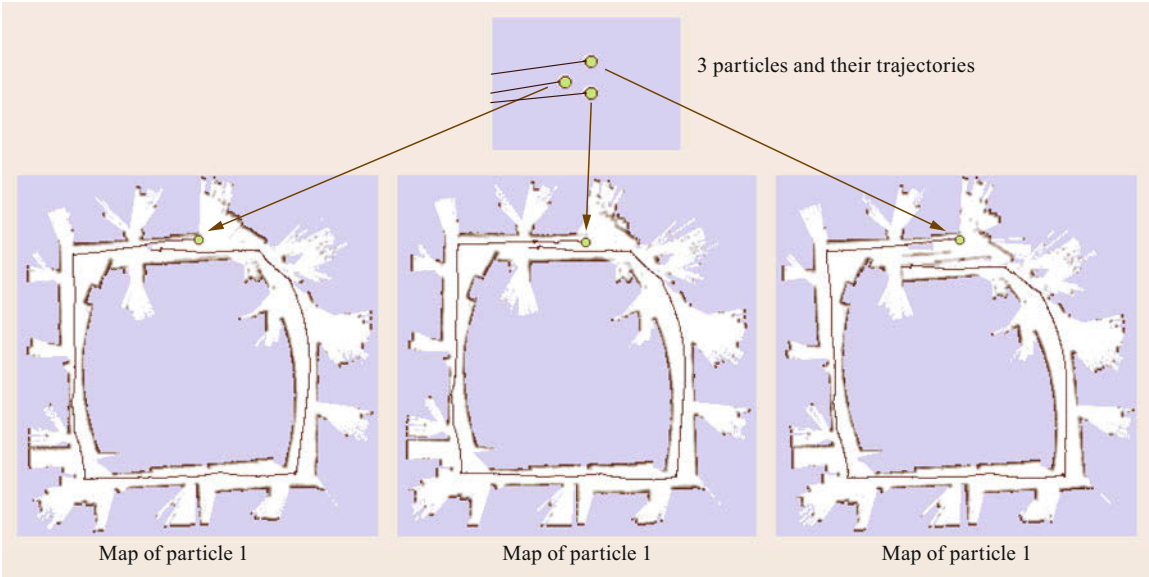


Fig. 46.6 Application of the grid-based variant of the FastSLAM algorithm. Each particle carries its own map and the importance weights of the particles are computed based on the likelihood of the measurements given the particle's own map

process for landmarks and grid maps, based on prior work in [46.52]. This leads to an improved sampling process

$$x_t^{[k]} \approx \frac{p(z_t \mid m_{t-1}^{[k]}, x_t)\, p(x_t \mid x_{t-1}^{[k]}, u_t)}{p(z_t \mid m_{t-1}^{[k]}, x_{t-1}^{[k]}, u_t)} \,, \qquad (46.14)$$

which incorporates the odometry and the observation at the same time. Using an improved proposal distribution leads to more accurately sampled locations. This in turn leads to more accurate maps and requires a smaller number of particles compared to approaches using the sampling process given in (46.12). This extension makes FastSLAM and especially its grid-based variants robust tools for addressing the SLAM problem.

Finally, there are approaches that aim to overcome the assumption that the observations show Gaussian characteristics. As shown in [46.47], there are several situations in which the model is nonGaussian and also multimodal. A sum of Gaussians model on a per-particle bases, however, can be efficiently considered in the particle filter and it eliminates this problem in practice without introducing additional computational demands.

The so-far developed particle filters-based SLAM systems suffer from two problems. First, the number of samples that are required to compute consistent maps is often set manually by making an *educated guess*. The larger the uncertainty that the filter needs to represent during mapping, the more critical becomes this parameter. Second, nested loops combined with extensive re-visits of previously mapped areas can lead to particle depletion, which in turn may prevent the system from estimating a consistent map. Adaptive resampling strategies [46.45], particles sharing maps [46.50], or filter backup approaches [46.53] improve the situation but cannot eliminate this problem in general.

46.2.3 Graph-Based Optimization Techniques

A third family of algorithms solves the SLAM problem through nonlinear sparse optimization. They draw their intuition from a graphical representation of the SLAM problem and the first working solution in robotics was proposed in [46.54]. The graph-based representation used here is closely related to a series of papers [46.55–64]. We note that most of the earlier techniques are offline and address the full SLAM problem. In more recent years, new incremental versions that effectively re-use the previously computed solution have been proposed such as [46.65–67].

The basic intuition of graph-based SLAM is a follows. Landmarks and robot locations can be thought of as nodes in a graph. Every consecutive pair of locations x_{t-1}, x_t is tied together by an edge that represents the information conveyed by the odometry reading u_t. Further edges exist between the nodes that correspond to locations x_t and landmarks m_i, assuming that at time t the robot sensed landmark i. Edges in this graph are

soft constraints. Relaxing these constraints yields the robot's best estimate for the map and the full path.

The construction of the graph is illustrated in Fig. 46.7. Suppose at time $t = 1$, the robot senses landmark m_1. This adds an arc in the (yet highly incomplete) graph between x_1 and m_1. When caching the edges in a matrix format (which happens to correspond to a quadratic equation defining the resulting constraints), a value is added to the elements between x_1 and m_1, as shown on the right hand side of Fig. 46.7a.

Now suppose the robot moves. The odometry reading u_2 leads to an arc between nodes x_1 and x_2, as shown in Fig. 46.7b. Consecutive application of these two basic steps leads to an graph of increasing size, as illustrated in Fig. 46.7c. Nevertheless this graph is *sparse*, in that each node is only connected to a small number of other nodes (assuming a sensor with limited sensing range). The number of constraints in the graph is (at worst) *linear* in the time elapsed and in the number of nodes in the graph.

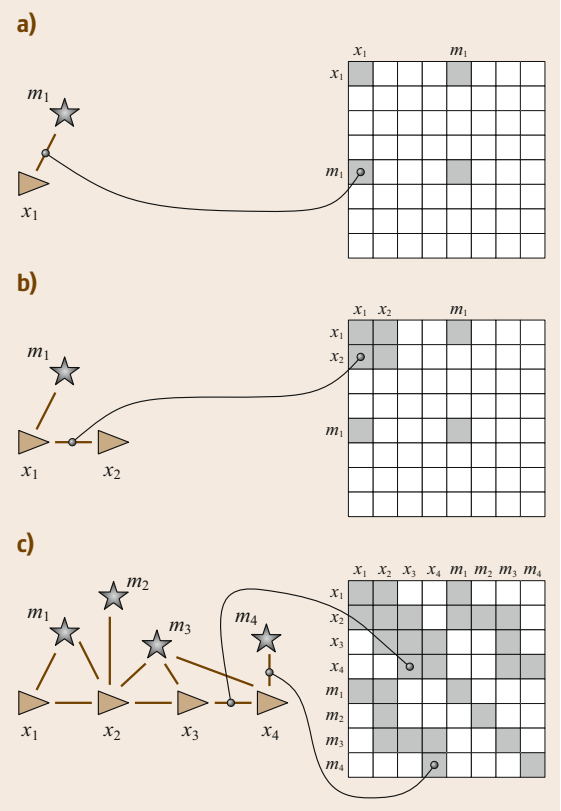


Fig.46.7a–c Illustration of the graph construction. The **(a)** diagram shows the graph, the **(b)** the constraints in matrix form. **(a)** Observation ls landmark m_1. **(b)** Robot motion from x_1 to x_2. **(c)** Several steps later

If we think of the graph as a spring-mass model [46.60], computing the SLAM solution is equivalent to computing the state of minimal energy of this model. To see, we note that the graph corresponds to the log-posterior of the full SLAM problem (46.4)

$$\log p(X_T, m \mid Z_T, U_T) \,. \tag{46.15}$$

Without derivation, we state that this logarithm is of the form

$$\begin{aligned} &\log p(X_T, m \mid Z_T, U_T) \\ &= \text{const} + \sum_t \log p(x_t \mid x_{t-1}, u_t) \\ &\quad + \sum_t \log p(z_t \mid x_t, m) \,, \end{aligned} \tag{46.16}$$

assuming independence between the individual observations and odometry readings. Each constraint of the form $\log p(x_t \mid x_{t-1}, u_t)$ is the result of exactly one robot motion event, and it corresponds to an edge in the graph. Likewise, each constraint of the form $\log p(z_t \mid x_t, m)$ is the result of one sensor measurement, to which we can also find a corresponding edge in the graph. The SLAM problem is then simply to find the mode of this equation, i. e.,

$$X_T^*, m^* = \underset{X_T, m}{\text{argmax}} \log p(X_T, m \mid Z_T, U_T) \,. \tag{46.17}$$

Without derivation, we note that under the Gaussian noise assumptions, which was made in the point-landmark example, this expression resolves to the following quadratic form

$$\begin{aligned} &\log p(X_T, m \mid Z_T, U_T) = \text{const} \\ &\quad + \sum_t \underbrace{[x_t - g(x_{t-1}, u_t)]^{\mathrm{T}} \, R_t^{-1} \, [x_t - g(x_{t-1}, u_t)]}_{\text{odometry reading}} \\ &\quad + \sum_t \underbrace{[z_t - h(x_t, m)]^{\mathrm{T}} \, Q_t^{-1} \, [z_t - h(x_t, m)]}_{\text{feature observation}} . \end{aligned} \tag{46.18}$$

This quadratic form yields a sparse system of equations and a number of efficient optimization techniques can be applied. Common choices include direct methods such as sparse Cholesky and QR decomposition, or iterative ones such as gradient descent, conjugate gradient, and others. Most SLAM implementations rely on iteratively linearizing the functions g and h, in which case the objective in (46.18) becomes quadratic in all of its variables.

Extensions to support an effective correction of large-scale graphs are hierarchical methods. One of

the first is the ATLAS framework [46.25], which constructs a two-level hierarchy combining a Kalman filter that operates in the lower level and a global optimization at the higher level. Similar to that, Hierarchical SLAM [46.68] is a technique for using independent local maps, which are merged in case of re-visiting a place. A fully hierarchical approach has been presented in [46.65]. It builds a multilevel pose-graph and employs an incremental, lazy optimization scheme that allows for optimizing large graphs and at the same time can be executed at each step during mapping. An alternative hierarchical approach is [46.69], which recursively partitions the graph into multiple-level submaps using the nested dissection algorithm.

When it comes to computing highly accurate environment reconstructions, approaches that do not only optimize the poses of the robot but also each individual measurement of a dense sensor often provide better results. In the spirit of bundle adjustment [46.4], approaches for laser scanners [46.70] and Kinect cameras [46.71] have been proposed.

The graphical paradigm can be extended to handle the data association problems as we can integrate additional knowledge on data association into (46.18). Suppose some oracle informed us that landmarks m_i and m_j in the map corresponded to one and the same physical landmark in the world. Then, we can either remove m_j from the graph and attach all adjacent edges to m_i, or we can add a soft correspondence constraint [46.72] of the form

$$(m_j - m_i)^{\mathrm{T}} \, \mathbf{\Gamma} \, (m_j - m_i) \; . \tag{46.19}$$

Here $\mathbf{\Gamma}$ is 2-by-2 diagonal matrix whose coefficients determine the penalty for *not* assigning identical locations to two landmarks (hence we want $\mathbf{\Gamma}$ to be large). Since graphical methods are usually used for the full SLAM problem, the optimization can be interleaved with the search for the optimal data association.

Data association errors typically have a strong impact in the resulting map estimate. Even a small number of wrong data associations is likely to result in inconsistent map estimates. Recently, novel approaches have been proposed that are robust under a certain number of false associations. For example, [46.73, 74] propose an iterative procedure that allows for disabling constraints, an action that is associated with a cost. A generalization of this method introduced in [46.75] formulates [46.74] as a robust cost function also reducing the computational requirements. Such approaches can deal with a significant number of false associations and still provide high-quality maps. Consistency checks for loop closure hypotheses can be found in other approaches as well, both in the front-end [46.76] and in the optimizer [46.77]. There has also been an extension that can deal with multimodal constraints [46.78], proposing a max-mixture representation for maintaining efficiency of the log likelihood optimizing in (46.16). As a result of that, the multimodal extension has only little impact on the runtime and can easily be incorporated in most optimizers. Also robust cost function are used for SLAM, for example pseudo Huber and several alternatives [46.75, 79–81].

Graphical SLAM methods have the advantage that they scale to much higher-dimensional maps than EKF SLAM, exploiting the sparsity of the graph. The key limiting factor in EKF SLAM is the covariance matrix, which takes space (and update time) quadratic in the size of the map. No such constraint exists in graphical methods. The update time of the graph is constant, and the amount of memory required is linear (under some mild assumptions). A further advantage of graph-based methods over the EKF is their ability to constantly relinearize the error function which often leads to better results. Performing the optimization can be expensive, however. Technically, finding the optimal data association is suspected to be an NP-hard problem, although in practice the number of plausible assignments is usually small. The continuous optimization of the log likelihood function in (46.18) depends among other things on the number and size of loops in the map. Also the initialization can have a strong impact on the result and a good initial guess can simplify the optimization substantially [46.8, 82, 83].

We note that the graph-based paradigm is very closely linked to information theory, in that the soft constraints constitute the information the robot has on the world (in an information-theoretic sense [46.92]). Most methods in the field are offline and they optimize for the entire robot path. If the robot path is long, the optimization may become cumbersome. Over the last five years, however, incremental optimization techniques have been proposed that aim at providing a sufficient but not necessarily perfect model of the environment at every point in time. This allows a robot to make decisions based on the current model, for example, to determine exploration goals. In this context, incremental variants [46.93, 94] of stochastic gradient descent techniques [46.8, 91] have been proposed that estimate which part of the graph requires re-optimization given new sensor data. Incremental methods [46.66, 79, 95] in the smoothing and mapping framework can be execute at each step of the mapping process and achieve the performance by variable ordering and selective relinearization. As also evaluated in [46.96] for the SLAM problem, variable ordering impacts the performance of the optimization. Others use hierarchical data structures [46.89] and pose-graphs [46.97] combined with

Table 46.1 Recent open-source graph-based SLAM implementations

Name	Comment
Dynamic covariance scaling (DCS) [46.75]	Optimization with a robust cost function for dealing with outliers Integrated into g^2o
g^2o [46.80]	Flexible and easily extendable optimization framework for SLAM Comes with different optimization approaches and error functions Supports external plugins
GTSAM2.1 [46.79]	Flexible optimization framework for SLAM and SFMstructure from motion Implements direct and iterative optimization techniques Implements smoothing and mapping (SAM), iSAM, and iSAM2 Implements bundle adjustment for Visual SLAM and SFM
HOG-Man [46.65]	Incremental optimization approach via hierarchical pose graphs and lazy optimization Requires pose-graphs with full rank constraints
iSAM2 [46.66]	General incremental nonlinear optimization with variable elimination Variable re-ordering to retain sparsity On-demand re-linearization of selected variables
KinFu (KinectFusion reimplemented)	Open source reimplementation of KinectFusion [46.84] within the point cloud library (PCL) Dense and highly accurate reconstruction using a Kinect camera Currently limited to medium sized rooms
MaxMixture [46.78]	Optimization for multimodal constraints and outliers Robust to outliers Plugin for g^2o
Parallel tracking and mapping (PTAM) [46.85]	System for tracking a hand-held monocular camera and observed features Operates on comparably small workspaces
RGBD-SLAM [46.86]	Kinect-frontend for HOG-Man and g^2o Fairly standard combination of SURF matching and RANSAC
ScaViSLAM [46.87]	SLAM system for stereo and Kinect-style cameras Combines local bundle adjustment with sparse global optimization for on-the-fly processing
SLAM6-D [46.88]	SLAM system that operates on point clouds from 3-D laser data Applies iterative closest point algorithm (ICP) and global relaxation
Sparse surface adjustment (SSA) [46.70, 71]	Optimizes robot poses and proximity sensor data jointly Provides smooth surface estimates Assumes a range sensor (e.g., laser scanner, Kinect, or similar)
TreeMap [46.89]	Incremental optimization approach Update in $O(\log N)$ time Provides only a mean estimate
TORO [46.90]	Optimization approach that extends stochastic gradient descent (SGD) [46.91] Robust under bad initial guesses Recovers quickly from large errors but slow convergence at minimum Assumes that constraints have roughly spherical covariance matrices Provides only a mean estimate
Vertigo [46.74]	Switchable constraints for robust optimization Plugin for g^2o

a lazy optimization for on-the-fly mapping [46.65]. As an alternative to global methods, relative optimization approaches [46.98] aim at computing locally consistent geometric maps but only topological maps on the global scale. Hybrid approaches [46.87] seek to combine the best of both worlds.

There also exists a number of cross-overs that manipulate the graph online so as to factor out past robot location variables. The resulting algorithms are filters [46.25, 33, 99, 100], and they tend to be intimately related to information filter methods. Many of the original attempts to decompose EKF SLAM representations into smaller submaps to scale up are based on motivations that are not dissimilar to the graphical approach [46.27, 28, 101].

Recently, researchers addressed the problem of long-term operation and frequent revisits of already mapped terrain. To avoid densely connected pose-graphs that lead to slow convergence behavior, the robot can switch between SLAM and localization, can merge nodes to avoid a growth of the graph [46.90, 102], or can discard nodes or edges [46.32, 103–105].

Graphical and optimization-based SLAM algorithm are still subject of intense research and the paradigm scales to maps large numbers of nodes [46.25, 55, 57, 59, 63–65, 89, 90, 106, 107]. Arguably, the graph-based

paradigm has generated some the largest SLAM maps ever built. Furthermore, the SLAM community started to release flexible optimization frameworks and SLAM implementations under open source licenses to support further developments and to allow for efficient comparisons, (Table 46.1). Especially the optimization frameworks [46.66, 79, 80] are flexible and powerful state of the art tools for developing graph-based SLAM systems. They can be either used as a *black box* or can be easily extended though plugins.

46.2.4 Relation of Paradigms

The three paradigms just discussed cover the vast majority of work in the field of SLAM. As discussed, EKF SLAM comes with a computational hurdle that poses serious scaling limitations and the linearization may lead to inconsistent maps. The most promising extensions of EKF SLAM are based on building local submaps; however, in many ways the resulting algorithms resemble the graph-based approach.

Particle filter methods sidestep some of the issues arising from the natural inter-feature correlations in the map – which hindered the EKF. By sampling from robot poses, the individual landmarks in the map become independent, and hence are decorrelated. As a result, FastSLAM can represent the posterior by a sampled robot pose, and many local, independent Gaussians for its landmarks. The particle representation offers advantages for SLAM as it allows for computationally efficient updates and for sampling over data associations. On the negative side, the number of necessary particles can grow very large, especially for robots seeking to map multiple nested loops.

Graph-based methods address the full SLAM problem, hence are in the standard formulation not online. They draw their intuition form the fact that SLAM can be modeled by a sparse graph of soft constraints, where each constraint either corresponds to a motion or a measurement event. Due to the availability of highly efficient optimization methods for sparse nonlinear optimization problems, graph-based SLAM has become the method of choice for building large-scale maps. Recent developments have brought up several graph-based methods for incremental map building that can be executed at every time step during navigation. Data association search can be incorporated into the basic mathematical framework and different approaches that are even robust under wrong data associations are available today.

46.3 Visual and RGB-D SLAM

A popular and important topic in recent years has been Visual SLAM – the challenge of building maps and tracking the robot pose in full 6-DOF using data from cameras [46.108] or RGB-D (Kinect) sensors [46.86, 109]. Visual sensors offer a wealth of information that enables the construction of rich 3-D models of the world. They also enable difficult issues such as loop-closing to be addressed in novel ways using appearance information [46.110]. Visual SLAM is anticipated to be a critical area for future research in perception for robotics, as we seek to develop low-cost systems that are capably of intelligent physical interaction with the world.

Attempting SLAM with monocular, stereo, omnidirectional, or RGB-D cameras raises the level-of-difficulty of many of the SLAM components, such as data association and computational efficiency, described above. A key challenge is robustness. Many visual SLAM applications of interest, such as augmented reality [46.85], entail handheld camera motions, which present greater difficulties for state estimation, in comparison to the motion of a wheeled robot across a flat floor.

Visual navigation and mapping was a key early goal in the mobile robotics community [46.111, 112], but early approaches were hampered by the lack of sufficient computational resources to handle massive video data streams. Early approaches were typically based on extended Kalman filters [46.113–116], but did not compute the full covariance for the feature poses and camera trajectory, resulting in a loss of consistency. Visual SLAM is closely related to the structure from motion (SFM) problem in computer vision [46.4, 5]. Historically, SFM was primarily concerned with off-line batch processing, whereas SLAM seeks to achieve a solution for online operation, suitable for closed-loop interaction of a robot or user with its environment. In comparison to laser scanners, cameras provide a fire hydrant of information, making online processing nearly impossible until recent increases in computation have become available.

Davison was an early pioneer in developing complete visual SLAM systems, initially using a real-time active stereo head [46.121] that tracked distinctive visual features with a full covariance EKF approach. Subsequent work developed the first real-time SLAM system that operated with a single freely moving camera as the only data source [46.23, 122]. This system could build sparse, room-size maps of indoor scenes at 30 Hz frame-rate in real-time, a notable historical achievement

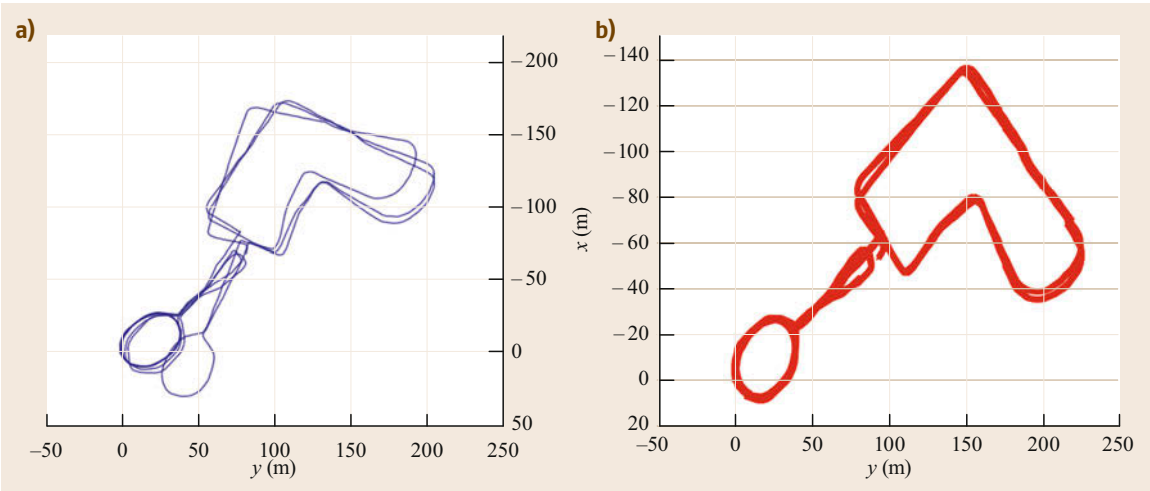


Fig. 46.8 **(a)** A 2-km path and 50 000 frames estimated for the New College Dataset (after [46.117]) using relative bundle adjustment (after [46.98]). **(b)** the relative bundle adjustment solution is easily improved by taking FAB-MAP (after [46.118]) loop-closures into account – this is achieved without global optimization (after [46.119]). Sibley et al. advocate that relative metric accuracy and topological consistency are the requirements for autonomous navigation, and these are better achieved using a relative manifold representation instead of using a conventional single Euclidean representation [46.120]

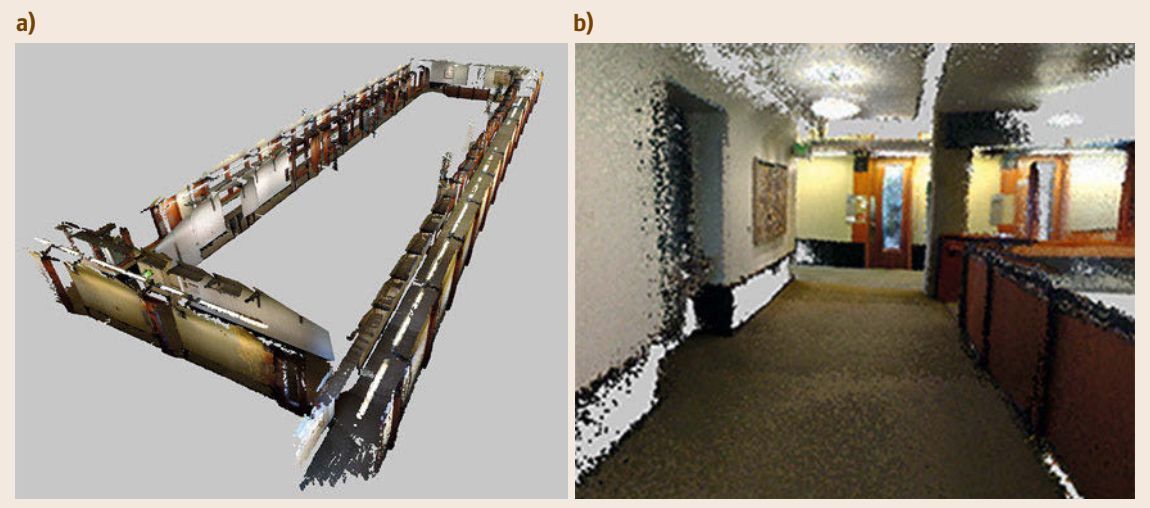


Fig. 46.9 **(a)** 3-D Model and **(b)** close-up view of a corridor environment in the Paul G. Allen building at University of Washington built from Kinect data (after [46.109]; image courtesy of Peter Henry, University of Washington)

in visual SLAM research. A difficulty encountered with initial monocular SLAM [46.23] was coping with the initialization of points that were far away from the camera, due to nonGaussian distributions of such feature locations resulting from poor depth information. This limitation was overcome in [46.24], introducing an inverse depth parameterization for monocular SLAM, a key development for enabling a unified treatment of initialization and tracking of visual features in real-time.

A milestone in creating robust visual SLAM systems was the introduction of keyframes in parallel tracking and mapping (PTAM) [46.85], which separated the tasks of keyframe mapping and localization into parallel threads, improving robustness and performance for online processing. Keyframes are now a mainstream concept for complexity reduction in visual SLAM systems. Related approaches using keyframes include [46.87, 123–126]. The work

in [46.108] analyzes the tradeoffs between filtering and keyframe-based bundle adjustment in visual SLAM, and concluded that keyframe bundle adjustment outperforms filtering, as it provides the most accuracy per unit of computing time.

As pointed out by Davison and other researchers, an appealing aspect of visual SLAM is that camera measurements can provide odometry information, and indeed visual odometry is a key component of modern SLAM systems [46.127]. Here, [46.128] and [46.129] provide an extensive tutorial of techniques for visual odometry, including feature detection, feature matching, outlier rejection, and constraint estimation, and trajectory optimization. Finally, a publicly available visual odometry library [46.130] that is optimized for efficient operation on small unmanned aerial vehicles is available today.

Visual information offers a tremendous source of information for loop closing, not present in the canonical 2-D laser SLAM systems developed in the early 2000s. The work in [46.110] was one of the first to employ techniques for visual object recognition [46.131] to location recognition. More recently FAB-MAP [46.118, 132] has demonstrated appearance-only place recognition at large scale, mapping trajectories with a length of 1000 km. Combining a bag-of-features approach with a probabilistic place model and Chow–Liu tree inference leads to place recognition that is robust against perceptual aliasing while remaining computationally efficient. Other work on place recognition includes [46.133], which combines bag-of-words loop closing with tests of geometrical consistency based on conditional random fields (Fig. 46.8).

The techniques described above have formed the basis for a number of notable large-scale SLAM systems developed in recent years. A 2008 special issue of the IEEE Transactions on Robotics provides a good snapshot of recent state-of-the-art SLAM techniques [46.135]. Other notable recent examples include [46.98, 119, 126, 136–138]. The idea of employing relative bundle adjustment [46.98] to compute a full maximum likelihood solution in an online fashion, even for loop closures, by employing a manifold representation that does not attempt to enforce Euclidean constraints results in maps that can be computed at high frame rate (see also Fig. 46.8). Finally, view-based mapping systems [46.126, 136, 137] aim at large-scale and/or life-long visual mapping based on the pose graph optimization techniques described above in Section 46.2.3.

Several compelling 3-D mapping and localization have been created in recent years with RGB-D (Kinect) sensors. The combination of direct range measurements with dense visual imagery can enable dramatic improvements in mapping and navigation systems for indoor environments. State-of-the-art RGB-D SLAM systems include [46.109] and [46.86]. Figure 46.9 shows examples of the output of these systems.

Other researchers aim at exploiting the surface properties of scanned environments to correct for sen-

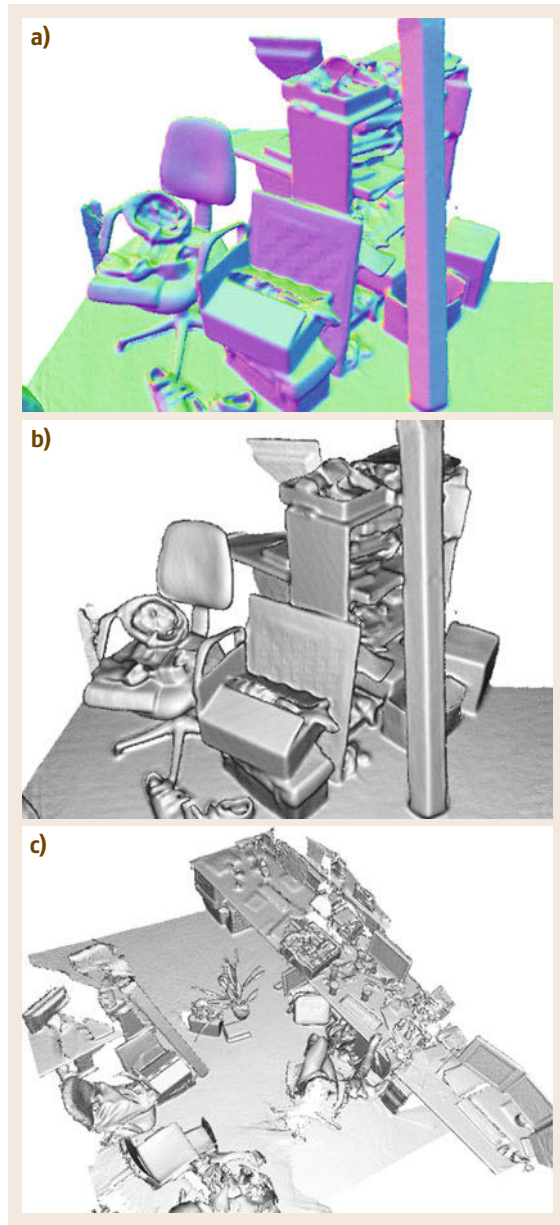


Fig.46.10a–c Results obtained with KinectFusion (after [46.134]). **(a)** A local scene as a normal map and **(b)** as a Phong-shaded rendering. The **(c)** image depicts a larger scene (image courtesy of Richard Newcombe, Imperial College London)

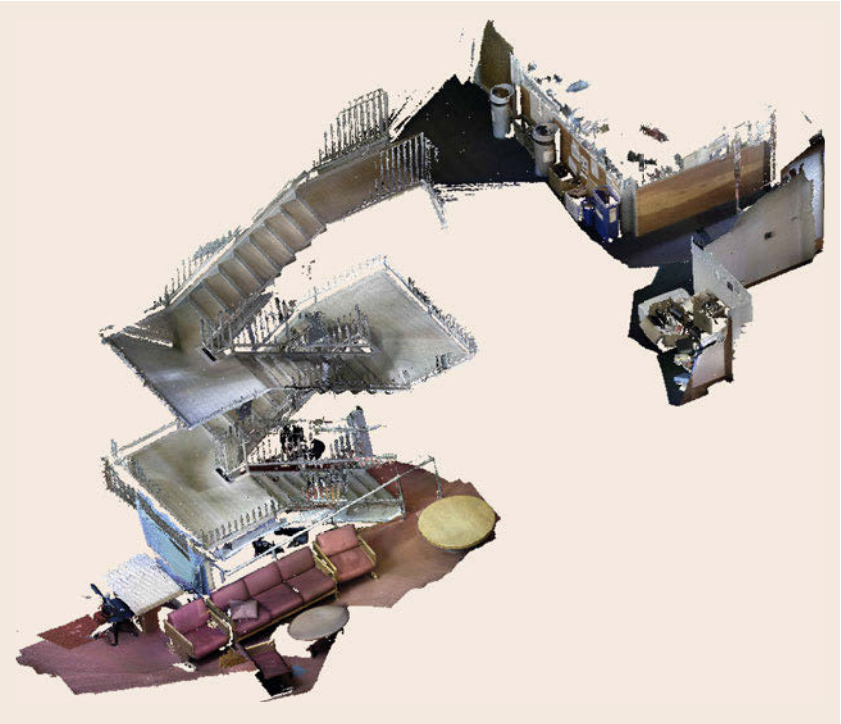


Fig. 46.11 Spatially extended KinectFusion output produced in real-time with Kintinuous (after [46.139])

sor noise of range sensors such as the Kinect [46.71]. They jointly optimize the poses of the sensor and the positions of the surface points measured and iteratively refine the structure of the error function by recomputing the data associations after each optimization, resulting in accurate smooth models of the environment.

An emerging area for future research is the development of fully dense processing methods that exploit recent advances in commodity graphical processing unit (GPU) technology. Kinect-based dense tracking and mapping, a fully-dense method for small-scale visual tracking and reconstruction is described in [46.140]. Dense modeling and tracking are achieved via highly parallelized operations on commodity GPU hardware to yield a system that outperforms previous methods such as PTAM for challenging camera trajectories. Dense methods offer an interesting perspective from which to address long-standing problems, such as visual odometry, from a fresh perspective, without requiring explicit feature detection and matching [46.141]. KinectFusion [46.84, 134] is a dense modeling system that tracks the 3-D pose of a handheld Kinect while concurrently reconstructing high-quality scene 3-D models in real-time. See Fig. 46.10 for an example. KinectFusion has been applied to spatially extended environments in [46.139, 142]. An example is shown in Fig. 46.11.

46.4 Conclusion and Future Challenges

This chapter has provided an introduction into SLAM, which is defined as the problem faced by a mobile platform roaming an unknown environment, and seeking to localize itself while concurrently building a map of the environment. The chapter discussed three main paradigms in SLAM, which are based on the extended Kalman filter, particle filters, and graph-based sparse optimization techniques, and then described recent progress in Visual/Kinect SLAM.

The following references provide an in-depth tutorial on SLAM and much greater depth of coverage on the details of popular SLAM algorithms. Furthermore, several implementations of popular SLAM systems, including most of the approaches listed in Table 46.1, can be found in online resources such as http://www.openslam.org or in the references [46.6, 9, 21, 62].

The considerable progress in SLAM in the past decade is beyond doubt. The core state estimation at

the heart of SLAM is now quite well understood, and a number of impressive implementations have been developed, including several widely used open source software implementations and some commercial projects. None-the-less, a number of open research challenges remain for the general problem of robotic mapping in complex and dynamic environments over extended periods of time, including robots sharing, extending, and revising previously built models, efficient failure recovery, zero user intervention, and operation on resource-constrained systems. Another exciting area for the future is the further development of fully dense visual mapping systems exploiting the latest advances in GPU hardware development.

An ultimate goal is to realize the challenge of *persistent* navigation and mapping – the capability for a robot to perform SLAM robustly for days, weeks, or months at a time with minimal human supervision, in complex and dynamic environments. Taking the limit as $t \to \infty$ poses difficult challenges to most current algorithms; in fact, most robot mapping and navigation algorithms are doomed to fail with the passage of time, as errors inevitably accrue. Despite recent encouraging solutions [46.102, 105], more research is needed for techniques that can recover from mistakes and enable robots to deal with changes in the environment and enabling a long-term autonomous existence.

Video-References

VIDEO 439 Deformation-based loop closure for Dense RGB-D SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/439

VIDEO 440 Large-scale SLAM using the Atlas framework
available from http://handbookofrobotics.org/view-chapter/46/videodetails/440

VIDEO 441 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/441

VIDEO 442 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/442

VIDEO 443 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/443

VIDEO 444 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/444

VIDEO 445 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/445

VIDEO 446 Graph-based SLAM using TORO
available from http://handbookofrobotics.org/view-chapter/46/videodetails/446

VIDEO 447 Sparse pose adjustment
available from http://handbookofrobotics.org/view-chapter/46/videodetails/447

VIDEO 449 Pose graph compression for laser-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/449

VIDEO 450 Pose graph compression for laser-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/450

VIDEO 451 Pose graph compression for laser-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/451

VIDEO 452 DTAM: Dense tracking and mapping in real-time
available from http://handbookofrobotics.org/view-chapter/46/videodetails/452

VIDEO 453 MonoSLAM: Real-time single camera SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/453

VIDEO 454 SLAM++: Simultaneous localisation and mapping at the level of objects
available from http://handbookofrobotics.org/view-chapter/46/videodetails/454

VIDEO 455 Extended Kalman filter SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/455

References

46.1 C.F. Gauss: *Theoria Motus Corporum Coelestium (Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections)* (Perthes and Bessen, Hamburg 1809), Republished in 1857 and by Dover in 1963

46.2 D.C. Brown: The bundle adjustment – Progress and prospects, Int. Arch. Photogramm. **21**(3), 3:3–3:35 (1976)

46.3 G. Konecny: *Geoinformation: Remote Sensing, Photogrammetry and Geographical Information Systems* (Taylor Francis, London 2002)

46.4 B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon: Bundle adjustment – A modern synthesis, Lect. Notes Comput. Sci. **62**, 298–372 (2000)

46.5 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2003)

46.6 T. Bailey, H.F. Durrant-Whyte: Simultaneous localisation and mapping (SLAM): Part II, Robotics Autom. Mag. **13**(3), 108–117 (2006)

46.7 H.F. Durrant-Whyte, T. Bailey: Simultaneous localisation and mapping (SLAM): Part I, Robotics Autom. Mag. **13**(2), 99–110 (2006)

46.8 G. Grisetti, C. Stachniss, W. Burgard: Nonlinear constraint network optimization for efficient map learning, IEEE Trans. Intell. Transp. Syst. **10**(3), 428–439 (2009)

46.9 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge, 2005)

46.10 R. Smith, M. Self, P. Cheeseman: A stochastic map for uncertain spatial relationships, Proc. Int. Symp. Robotics Res. (ISRR) (MIT Press, Cambridge 1988) pp. 467–474

46.11 R. Smith, M. Self, P. Cheeseman: Estimating uncertain spatial relationships in robotics. In: *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer Verlag, Berlin, Heidelberg 1990) pp. 167–193

46.12 P. Moutarlier, R. Chatila: Stochastic multisensory data fusion for mobile robot location and environment modeling, 5th Int. Symp. Robotics Res. (ISRR) (1989) pp. 207–216

46.13 P. Moutarlier, R. Chatila: An experimental system for incremental environment modeling by an autonomous mobile robot, 1st Int. Sym. Exp. Robotics (ISER) (1990)

46.14 R. Kalman: A new approach to linear filtering and prediction problems, J. Fluids **82**, 35–45 (1960)

46.15 A.M. Jazwinsky: *Stochastic Processes and Filtering Theory* (Academic, New York 1970)

46.16 M.G. Dissanayake, P.M. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba: A solution to the simultaneous localization and map building (SLAM) Problem, IEEE Trans. Robotics Autom. **17**(3), 229–241 (2001)

46.17 J. Neira, J. Tardos, J. Castellanos: Linear time vehicle relocation in SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 427–433

46.18 J. Neira, J.D. Tardos: Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robotics Autom. **17**(6), 890–897 (2001)

46.19 G.D. Tipaldi, M. Braun, K.O. Arras: Flirt: interest regions for 2D range data with applications to robot navigation, Proc. Int. Symp. Exp. Robotics (ISER) (2010)

46.20 G.D. Tipaldi, L. Spinello, W. Burgard: Geometrical flirt phrases for large scale place recognition in 2D range data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.21 T. Bailey: Mobile Robot Localisation and Mapping in Extensive Outdoor Environments, Ph.D. Thesis (Univ. of Sydney, Sydney 2002)

46.22 J.J. Leonard, R.R. Rikoski, P.M. Newman, M. Bosse: Mapping partially observable features from multiple uncertain vantage points, Int. J. Robotics Res. **21**(10), 943–975 (2002)

46.23 A.J. Davison: Real-time simultaneous localisation and mapping with a single camera, Proc. IEEE 9th Int. Conf. Comput. Vis. (2003) pp. 1403–1410

46.24 J.M.M. Montiel, J. Civera, A.J. Davison: Unified inverse depth parametrization for monocular SLAM, Robotics Sci. Syst., Vol. 1 (2006)

46.25 M. Bosse, P.M. Newman, J. Leonard, S. Teller: Simultaneous localization and map building in large-scale cyclic environments using the Atlas Framework, Int. J. Robotics Res. **23**(12), 1113–1139 (2004)

46.26 J. Nieto, T. Bailey, E. Nebot: Scan-SLAM: Combining EKF-SLAM and scan correlation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

46.27 J. Guivant, E. Nebot: Optimization of the simultaneous localization and map building algorithm for real time implementation, IEEE Trans. Robotics. Autom. **17**(3), 242–257 (2001)

46.28 J.J. Leonard, H. Feder: A computationally efficient method for large-scale concurrent mapping and localization, Proc. 9th Int. Symp. Robotics Res. (ISRR), ed. by J. Hollerbach, D. Koditschek (1999) pp. 169–176

46.29 J.D. Tardós, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, Int. J. Robotics Res. **21**(4), 311–330 (2002)

46.30 S.B. Williams, G. Dissanayake, H.F. Durrant-Whyte: Towards multi-vehicle simultaneous localisation and mapping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2743–2748

46.31 R.M. Eustice, H. Singh, J.J. Leonard: Exactly sparse delayed-state filters for view-based SLAM, IEEE Trans. Robotics **22**(6), 1100–1114 (2006)

46.32 V. Ila, J.M. Porta, J. Andrade-Cetto: Information-based compact pose SLAM, IEEE Trans. Robotics **26**(1), 78–93 (2010)

46.33 S. Thrun, D. Koller, Z. Ghahramani, H.F. Durrant-Whyte, A.Y. Ng: Simultaneous mapping and localization with sparse extended information filters, Proc.5th Int. Workshop Algorithmic Found. Robotics, ed. by J.-D. Boissonnat, J. Burdick, K. Goldberg, S. Hutchinson (2002)

Avidbots Ex1016
Page 1199 of 2154

46.34 M.R. Walter, R.M. Eustice, J.J. Leonard: Exactly sparse extended information filters for feature-based SLAM, Int. J. Robotics Res. **26**(4), 335–359 (2007)

46.35 T. Bailey, J. Nieto, J. Guivant, M. Stevens, E. Nebot: Consistency of the EKF-SLAM algorithm, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 3562–3568

46.36 S. Huang, G. Dissanayake: Convergence and consistency analysis for extended Kalman filter based SLAM, IEEE Trans. Robotics **23**(5), 1036–1049 (2007)

46.37 G.P. Huang, A.I. Mourikis, S.I. Roumeliotis: Observability-based Rules for Designing Consistent EKF SLAM Estimators, Int. J. Robotics Res. **29**, 502–528 (2010)

46.38 N. Metropolis, S. Ulam: The Monte Carlo method, J. Am. Stat. Assoc. **44**(247), 335–341 (1949)

46.39 D. Blackwell: Conditional expectation and unbiased sequential estimation, Ann. Math. Stat. **18**, 105–110 (1947)

46.40 C.R. Rao: Information and accuracy obtainable in estimation of statistical parameters, Bull. Calcutta Math. Soc. **37**(3), 81–91 (1945)

46.41 K. Murphy, S. Russel: Rao–Blackwellized particle filtering for dynamic Bayesian networks. In: *Sequential Monte Carlo Methods in Practice*, ed. by A. Docout, N. de Freitas, N. Gordon (Springer, Berlin 2001) pp. 499–516

46.42 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, Proc. AAAI Natl. Conf. Artif. Intell. (2002)

46.43 J. Pearl: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo 1988)

46.44 J. Guivant, E. Nebot, S. Baiker: Autonomous navigation and map building using laser range sensors in outdoor applications, J. Robotics Syst. **17**(10), 565–583 (2000)

46.45 G. Grisetti, C. Stachniss, W. Burgard: Improved techniques for grid mapping with Rao-Blackwellized particle filters, IEEE Trans. Robotics **23**, 34–46 (2007)

46.46 D. Hähnel, D. Fox, W. Burgard, S. Thrun: A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003)

46.47 C. Stachniss, G. Grisetti, W. Burgard, N. Roy: Evaluation of gaussian proposal distributions for mapping with rao-blackwellized particle filters, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)

46.48 A. Eliazar, R. Parr: DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, Proc. 16th Int. Jt. Conf. Artif. Intell. (IJCAI) (2003) pp. 1135–1142

46.49 A. Eliazar, R. Parr: DP-SLAM 2.0, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2004) pp. 1314–1320

46.50 G. Grisetti, G.D. Tipaldi, C. Stachniss, W. Burgard, D. Nardi: Fast and accurate SLAM with Rao-Blackwellized particle filters, J. Robotics Auton. Syst. **55**(1), 30–38 (2007)

46.51 D. Roller, M. Montemerlo, S. Thrun, B. Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, Int. Jt. Conf. Artif. Intell. (IICAI) (Morgan Kaufmann, San Francisco 2003) pp. 1151–1156

46.52 R. van der Merwe, N. de Freitas, A. Doucet, E. Wan: The unscented particle filter, Proc. Adv. Neural Inform. Process. Syst. Conf. (2000) pp. 584–590

46.53 C. Stachniss, G. Grisetti, W. Burgard: Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 655–660

46.54 F. Lu, E. Milios: Globally consistent range scan alignment for environmental mapping, Auton. Robots **4**, 333–349 (1997)

46.55 F. Dellaert: Square root SAM, Robotics Sci. Syst., ed. by S. Thrun, G. Sukhatme, S. Schaal, O. Brock (MIT Press, Cambridge 2005)

46.56 T. Duckett, S. Marsland, J. Shapiro: Learning globally consistent maps by relaxation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 3841–3846

46.57 T. Duckett, S. Marsland, J. Shapiro: Fast, online learning of globally consistent maps, Auton. Robots **12**(3), 287–300 (2002)

46.58 J. Folkesson, H.I. Christensen: Graphical SLAM: A self-correcting map, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 383–390

46.59 U. Frese, G. Hirzinger: Simultaneous localization and mapping – A discussion, Proc. IJCAI Workshop Reason. Uncertain. Robotics (2001) pp. 17–26

46.60 M. Golfarelli, D. Maio, S. Rizzi: Elastic correction of dead-reckoning errors in map building, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 905–911

46.61 J. Gutmann, K. Konolige: Incremental mapping of large cyclic environments, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (CIRA) (2000) pp. 318–325

46.62 G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard: A Tutorial on Graph-based SLAM, IEEE Trans. Intell. Transp. Syst. Mag. **2**, 31–43 (2010)

46.63 K. Konolige: Large-scale map-making, Proc. AAAI Natl. Conf. Artif. Intell. (MIT Press, Cambridge 2004) pp. 457–463

46.64 M. Montemerlo, S. Thrun: Large-scale robotic 3-D mapping of urban structures, Springer Tract. Adv. Robotics **21**, 141–150 (2005)

46.65 G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, C. Hertzberg: Hierarchical optimization on manifolds for online 2D and 3D mapping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

46.66 M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, F. Dellaert: iSAM2: Incremental smoothing and mapping using the Bayes tree, Int. J. Robotics Res. **31**, 217–236 (2012)

46.67 M. Kaess, A. Ranganathan, F. Dellaert: iSAM: Incremental Smoothing and Mapping, IEEE Trans. Robotics **24**(6), 1365–1378 (2008)

Avidbots Ex1016
Page 1200 of 2154

46.68 C. Estrada, J. Neira, J.D. Tardós: Hierarchical SLAM: Real-time accurate mapping of large environments, IEEE Trans. Robotics **21**(4), 588–596 (2005)

46.69 K. Ni, F. Dellaert: Multi-level submap based SLAM using nested dissection, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

46.70 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

46.71 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate 3D surface models by sparse surface adjustment, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

46.72 Y. Liu, S. Thrun: Results for outdoor-SLAM using sparse extended information filters, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

46.73 N. Sünderhauf, P. Protzel: BRIEF-Gist – Closing the loop by simple means, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 1234–1241

46.74 N. Sünderhauf, P. Protzel: Switchable constraints for robust pose graph SLAM, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

46.75 P. Agarwal, G.D. Tipaldi, L. Spinello, C. Stachniss, W. Burgard: Robust map optimization using dynamic covariance scaling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.76 E. Olson: Recognizing places using spectrally clustered local matches, J. Robotics Auton. Syst. **57**(12), 1157–1172 (2009)

46.77 Y. Latif, C. Cadena Lerma, J. Neira: Robust loop closing over time, Robotics Sci. Syst. (2012)

46.78 E. Olson, P. Agarwal: Inference on networks of mixtures for robust robot mapping, Robotics Sci. Syst. (2012)

46.79 F. Dellaert: *Factor graphs and GTSAM: A hands-on introduction*, Tech. Rep. GT-RIM-CP & R-2012-002 (Georgia Tech, Atlanta 2012)

46.80 R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard: G^2o: A general framework for graph optimization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

46.81 D.M. Rosen, M. Kaess, J.J. Leonard: An incremental trust-region method for robust online sparse least-squares estimation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1262–1269

46.82 L. Carlone, R. Aragues, J. Castellanos, B. Bona: A linear approximation for graph-based simultaneous localization and mapping, Robotics Sci. Syst. (2011)

46.83 G. Grisetti, R. Kümmerle, K. Ni: Robust optimization of factor graphs by using condensed measurements, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

46.84 S. Izadi, R.A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A.J. Davison, A. Fitzgibbon: Kinectfusion: Real-time dynamic 3D surface reconstruction and interaction, ACM SIGGRAPH Talks (2011) p. 23

46.85 G. Klein, D. Murray: Parallel tracking and mapping for small AR workspaces, IEEE ACM Int. Symp. Mixed Augment. Real. (ISMAR) (2007) pp. 225–234

46.86 F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard: An evaluation of the RGB-D SLAM system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

46.87 H. Strasdat, A.J. Davison, J.M.M. Montiel, K. Konolige: Double window optimisation for constant time visual SLAM, Int. Conf. Computer Vis. (ICCV) (2011)

46.88 A. Nüchter: 3D robot mapping, Springer Tract. Adv. Robotics **52** (2009)

46.89 U. Frese: Treemap: An $O(\log n)$ algorithm for indoor simultaneous localization and mapping, Auton. Robots **21**(2), 103–122 (2006)

46.90 G. Grisetti, C. Stachniss, S. Grzonka, W. Burgard: A tree parameterization for efficiently computing maximum likelihood maps using gradient descent, Robotics Sci. Syst. (2007)

46.91 E. Olson, J.J. Leonard, S. Teller: Fast iterative alignment of pose graphs with poor initial estimates, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2262–2269

46.92 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)

46.93 E. Olson, J.J. Leonard, S. Teller: Spatially-adaptive learning rates for online incremental SLAM, Robotics Sci. Syst. (2007)

46.94 G. Grisetti, D. Lordi Rizzini, C. Stachniss, E. Olson, W. Burgard: Online constraint network optimization for efficient maximum likelihood map learning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008)

46.95 M. Kaess, A. Ranganathan, F. Dellaert: Fast incremental square root information smoothing, Int. Jt. Conf. Artif. Intell. (ISCAI) (2007)

46.96 P. Agarwal, E. Olson: Evaluating variable reordering strategies for SLAM, Proc. IEEE/RSJ Int. Conf. Intel Robots Syst. (IROS) (2012)

46.97 K. Ni, D. Steedly, F. Dellaert: Tectonic SAM: exact, out-of-core, submap-based SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 1678–1685

46.98 G. Sibley, C. Mei, I. Reid, P. Newman: Adaptive relative bundle adjustment, Robotics Sci. Syst. (2009)

46.99 P.M. Newman, J.J. Leonard, R. Rikoski: Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar, 11th Int. Symp. Robotics Res. (2003)

46.100 M.A. Paskin: Thin junction tree filters for simultaneous localization and mapping, Int. Jt. Conf. Artif. Intell. (IJCAI) (Morgan Kaufmann, New York 2003) pp. 1157–1164

46.101 S.B. Williams: Efficient Solutions to Autonomous Mapping and Navigation Problems, Ph.D. Thesis (ACFR Univ. Sydney, Sydney 2001)

46.102 H. Johannsson, M. Kaess, M.F. Fallon, J.J. Leonard: Temporally scalable visual SLAM using a reduced pose graph, RSS Workshop Long-term Oper. Auton. Robotic Syst. Chang. Environ. (2012)

Avidbots Ex1016
Page 1201 of 2154

46.103 N. Carlevaris-Bianco, R.M. Eustice: Generic factor-based node marginalization and edge sparsification for pose-graph SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.104 M. Kaess, F. Dellaert: Covariance recovery from a square root information matrix for data association, J. Robotics Auton. Syst. **57**(12), 1198–1210 (2009)

46.105 H. Kretzschmar, C. Stachniss: Information-theoretic compression of pose graphs for laser-based SLAM, Int. J. Robotics Res. **31**(11), 1219–1230 (2012)

46.106 U. Frese, L. Schröder: Closing a million-landmarks loop, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

46.107 J. McDonald, M. Kaess, C. Cadena, J. Neira, J.J. Leonard: Real-time 6-DOF multi-session visual SLAM over large scale environments, J. Robotics Auton. Syst. **61**(10), 1144–1158 (2012)

46.108 H. Strasdat, J.M.M. Montiel, A.J. Davison: Real-time monocular SLAM: Why filter?, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

46.109 P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox: RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments, Int. J. Robotics Res. **31**(5), 647–663 (2012)

46.110 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (ICCVPR) (2006) pp. 2161–2168

46.111 R.A. Brooks: Aspects of mobile robot visual map making, Proc. Int. Symp. Robotics Res. (ISRR) (MIT Press, Cambridge 1984) pp. 287–293

46.112 H. Moravec: Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover, Ph.D. Thesis (Stanford Univ., Stanford 1980)

46.113 N. Ayache, O. Faugeras: Building, registrating, and fusing noisy visual maps, Int. J. Robotics Res. **7**(6), 45–65 (1988)

46.114 D. Kriegman, E. Triendl, T. Binford: Stereo vision and navigation in buildings for mobile robots, IEEE Trans. Robotics Autom. **5**(6), 792–803 (1989)

46.115 L. Matthies, S. Shafer: Error modeling in stereo navigation, IEEE J. Robotics Autom. **3**(3), 239–248 (1987)

46.116 S. Pollard, J. Porrill, J. Mayhew: Predictive feedforward stereo processing, Alvey Vis. Conf. (1989) pp. 97–102

46.117 M. Smith, I. Baldwin, W. Churchill, R. Paul, P. Newman: The new college vision and laser data set, Int. J. Robotics Res. **28**(5), 595–599 (2009)

46.118 M. Cummins, P.M. Newman: Appearance-only SLAM at large scale with FAB-MAP 2.0, Int. J. Robotics Res. **30**(9), 1100–1123 (2010)

46.119 P.M. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schroter, L. Murphy, W. Churchill, D. Cole, I. Reid: Navigating, recognising and describing urban spaces with vision and laser, Int, J. Robotics Res. **28**, 11–12 (2009)

46.120 G. Sibley, C. Mei, I. Reid, P. Newman: Vast-scale outdoor navigation using adaptive relative bundle adjustment, Int. J. Robotics Res. **29**(8), 958–980 (2010)

46.121 A. Davison, D. Murray: Mobile robot localisation using active vision, Eur. Conf. Comput. Vis. (ECCV) (1998) pp. 809–825

46.122 A.J. Davison, I. Reid, N. Molton, O. Stasse: MonoSLAM: Real-time single camera SLAM, IEEE Trans., Pattern Anal. Mach. Intell. **29**(6), 1052–1067 (2007)

46.123 R.O. Castle, G. Klein, D.W. Murray: Wide-area augmented reality using camera tracking and mapping in multiple regions, Comput. Vis. Image Underst. **115**(6), 854–867 (2011)

46.124 E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular SLAM, Br. Mach. Vis. Conf. (2008)

46.125 E. Eade, P. Fong, M.E. Munich: Monocular graph SLAM with complexity reduction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3017–3024

46.126 K. Konolige, M. Agrawal: FrameSLAM: From bundle adjustment to real-time visual mapping, IEEE Trans. Robotics **24**(5), 1066–1077 (2008)

46.127 D. Nister, O. Naroditsky, J. Bergen: Visual odometry for ground vehicle applications, J. Field Robotics **23**(1), 3–20 (2006)

46.128 D. Scaramuzza, F. Fraundorfer: Visual odometry. Part I: The first 30 years and fundamentals, IEEE Robotics Autom. Mag. **18**(4), 80–92 (2011)

46.129 F. Fraundorfer, D. Scaramuzza: Visual odometry. Part II: Matching, robustness, optimization, and applications, IEEE Robotics Autom. Mag. **19**(2), 78–90 (2012)

46.130 A.S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, N. Roy: Visual odometry and mapping for autonomous flight using an RGB-D camera, Proc. Int. Symp. Robotics Res. (ISRR) (2011)

46.131 J. Sivic, A. Zisserman: Video Google: A text retrieval approach to object matching in videos, Int. Conf. Computer Vis. (ICCV) (2003) p. 1470

46.132 M. Cummins, P.M. Newman: Probabilistic appearance based navigation and loop closing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2042–2048

46.133 C. Cadena, D. Gálvez, F. Ramos, J.D. Tardós, J. Neira: Robust place recognition with stereo cameras, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

46.134 R.A. Newcombe, A.J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, A. Fitzgibbon: Kinectfusion: Real-time dense surface mapping and tracking, IEEE/ACM Int. Sym. Mixed Augment. Real. (ISMAR) (2011) pp. 127–136

46.135 J. Neira, A.J. Davison, J.J. Leonard: Guest editorial special issue on visual SLAM, IEEE Trans. Robotics **24**(5), 929–931 (2008)

46.136 K. Konolige, J. Bowman: Towards lifelong visual maps, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1156–1163

46.137 K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, P. Fua: View-

based maps, Int. J. Robotics Res. **29**(8), 941–957 (2010)
46.138 G. Sibley, C. Mei, I. Reid, P. Newman: Planes, trains and automobiles – Autonomy for the modern robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 285–292
46.139 T. Whelan, H. Johannsson, M. Kaess, J.J. Leonard, J.B. McDonald: Robust real-time visual odometry for dense RGB-D mapping, IEEE Int. Conf. Robotics Autom. (ICRA (2013)
46.140 R.A. Newcombe, S.J. Lovegrove, A.J. Davison: DTAM: Dense tracking and mapping in real-time, Int. Conf. Computer Vis. (ICCV) (2011) pp. 2320–2327
46.141 F. Steinbruecker, J. Sturm, D. Cremers: Real-time visual odometry from dense RGB-D images, Workshop Live Dense Reconstr. Mov. Cameras Int. Conf. Comput. Vis. (ICCV) (2011)
46.142 H. Roth, M. Vona: Moving volume kinectfusion, Br. Mach. Vis. Conf. (2012)

Avidbots Ex1016
Page 1203 of 2154

Multimedia Contents

47. Motion Planning and Obstacle Avoidance

Javier Minguez, Florant Lamiraux, Jean-Paul Laumond

This chapter describes motion planning and obstacle avoidance for mobile robots. We will see how the two areas do not share the same modeling background. From the very beginning of motion planning, research has been dominated by computer sciences. Researchers aim at devising well-grounded algorithms with well-understood completeness and exactness properties.

The challenge of this chapter is to present both nonholonomic motion planning (Sects. 47.1–47.6) and obstacle avoidance (Sects. 47.7–47.10) issues. Section 47.11 reviews recent successful approaches that tend to embrace the whole problem of motion planning and motion control. These approaches benefit from both nonholonomic motion planning and obstacle avoidance methods.

47.1 **Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory** 1178
47.2 **Kinematic Constraints and Controllability** 1179
47.2.1 Definitions 1179
47.2.2 Controllability 1179
47.2.3 Example: The Differentially-Driven Mobile Robot 1180
47.3 **Motion Planning and Small-Time Controllability** 1180
47.3.1 The Decision Problem 1180
47.3.2 The Complete Problem 1181
47.4 **Local Steering Methods and Small-Time Controllability** 1181
47.4.1 Local Steering Methods Accounting for Small-Time Controllability 1182
47.4.2 Equivalence Between Chained-Formed and Feedback-Linearizable Systems 1184
47.5 **Robots and Trailers** 1184
47.5.1 Differentially-Driven Mobile Robots 1184
47.5.2 Differentially-Driven Mobile Robots Towing One Trailer 1184
47.5.3 Car-Like Mobile Robots 1185
47.5.4 Bi-Steerable Mobile Robots 1185
47.5.5 Differentially-Driven Mobile Robots Towing Trailers 1185
47.5.6 Open Problems 1186
47.6 **Approximate Methods** 1186
47.6.1 Forward Dynamic Programming 1186
47.6.2 Discretization of the Input Space 1186
47.6.3 Input-Based Rapidly Exploring Random Trees 1187
47.7 **From Motion Planning to Obstacle Avoidance** 1187
47.8 **Definition of Obstacle Avoidance** 1187
47.8.1 Obstacle Avoidance Problem 1187
47.9 **Obstacle Avoidance Techniques** 1188
47.9.1 Potential Field Methods 1189
47.9.2 Vector Field Histogram 1190
47.9.3 The Obstacle Restriction Method 1191
47.9.4 Dynamic Window Approach 1192
47.9.5 Velocity Obstacles 1192
47.9.6 Nearness Diagram Navigation ... 1193
47.10 **Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance** 1194
47.10.1 Techniques that Abstract Vehicle Aspects 1194
47.10.2 Techniques of Decomposition in Subproblems 1195

47.11 Integration Planning – Reaction 1196
47.11.1 Systems of Path Deformation 1196
47.11.2 Systems of Tactical Planning 1197
47.12 Conclusions, Future Directions, and Further Reading 1198
Video-References 1199
References 1199

The introduction of nonholonomic constraints has entailed revisiting these algorithms via the introduction of differential geometry materials. Such a combination has been made possible for certain classes of systems, the so-called small-time controllable ones. It remains that the underlying hypothesis of motion planning algorithms is the knowledge of a global and accurate map of the environment. More than that the considered system is a formal system of equations that does not account for the entire physical system: uncertainties in the world or system modeling are not considered. Such hypotheses are too strong in practice. This is why other complementary research has been done in parallel in a more pragmatic but realistic manner. Such research deals with obstacle avoidance. The problem here is not to deal with complicated systems like a car with multiple trailers. The considered systems are much simpler with respect to their geometric shape. The problem considers sensor-based motions to face the physical issues of a real system navigating in a real world better than motion planning algorithms. How can we *navigate* toward a goal in a cluttered environment when the obstacles to avoid have just been discovered in real time? This is the question that obstacle avoidance addresses.

The appearance of mobile robots in the late 1960s early 1970s initiated a new research domain: autonomous navigation. It is interesting to note that the first navigation systems were published at the very first International Joint Conferences on Artificial Intelligence (IJCAI 1969). These systems were based on seminal ideas, which have been very fruitful in the development of robot motion planning algorithms. For instance, in 1969, the mobile robot Shakey used a grid-based approach to model and explore the environment [47.1], in 1977 Jason used a visibility graph built from the corners of the obstacles [47.2], and in 1979 Hilare decomposed the environment into collision-free convex cells [47.3].

In the late 1970s, studies of robot manipulators popularized the notion of the configuration space of a mechanical system [47.4]. In the configuration space the *piano* becomes a point. The motion planning problem for a mechanical system was thus reduced to finding a path for a point in the configuration space. The way was open to extending the seminal ideas and to developing new and well-grounded algorithms (see *Latombe*'s book [47.5]).

One decade later, the notion of nonholonomic systems (also borrowed from mechanics) appeared in the literature [47.6] on robot motion planning through the problem of car parking. This problem had not been solved by the pioneering works on mobile robot navigation. Nonholonomic motion planning then became an attractive research field [47.7].

Besides this research effort in path planning, work was initiated in order to make robots move out of their initially artificial environments where the world was cylindrical and composed of wooden vertical boards. Robots started to move in laboratory buildings, with people walking around. Inaccurate localization, uncertain and incomplete maps of the world, and unexpected moving or static obstacles made roboticists aware of the gap between planning a path and executing a motion. Since then the domain of obstacle avoidance has been very active.

In the 2000s a lot of effort has been made toward the integration of motion planning and obstacle avoidance. This effort was stimulated by the DARPA Urban challenge competition. The most successful integration was done by the Carnegie Mellon University team, who won the competition [47.8].

47.1 Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory

Nonholonomic constraints are nonintegrable linear constraints over the velocity space of a system. For instance, the rolling without slipping constraint of a differentially-driven mobile robot (Fig. 47.1) is linear with respect to the velocity vector (vector of linear and angular velocities) of the differentially-driven robot and is non integrable (it cannot be integrated into a constraint over the configuration variables). As a consequence, a differentially-driven mobile robot can go anywhere but not following any trajectory. Other types of nonholonomic constraints arise when considering second-order differential equations such as the conser-

vation of inertial momentum. A number of papers investigate the famous falling cat problem or free-floating robots in space [47.7]. This chapter is devoted to nonholonomic constraints for mobile robots with wheels.

While the constraints due to obstacles are expressed directly in the configuration space, that is a manifold, nonholonomic constraints are expressed in the tangent space. In the presence of a linear kinematic constraint, the first question that naturally arises is: *does this constraint reduce the space* reachable *by the system?* This question can be answered by studying the structure of the distribution spanned by the Lie algebra of the control system.

Even in the absence of obstacles, planning admissible motions (i. e., that satisfy the kinematic constraints) for a nonholonomic system between two configurations is not an easy task. Exact solutions have been proposed only for some classes of systems but a lot of systems remain without exact solution. In the general case, however, approximate solutions can be used.

The motion planning problem for a nonholonomic system can be stated as follows: given a map of the environment with obstacles in the workspace, a robot subject to nonholonomic constraints, an initial configuration and a goal configuration, find an admissible collision-free path between the initial and goal configurations. Solving this problem requires taking into account both the configuration space constraints due to obstacles and the nonholonomic constraints. The tools developed to address this issue thus combine motion planning and control theory techniques. Such a combination is possible for the class of so-called small-time controllable systems due to topological arguments (Theorem 47.2 in Sect. 47.3).

47.2 Kinematic Constraints and Controllability

In this section, we give the main definition of controllability, using *Sussman*'s terminology [47.9].

47.2.1 Definitions

Let us denote by CS the configuration space of dimension n of a given mobile robot and by $\boldsymbol{q}$ the configuration of this robot. If the robot is mounted on wheels, it is subject to kinematic constraints, linear in the velocity vector

$$\omega_i(\boldsymbol{q})\dot{\boldsymbol{q}} = 0\,, \quad i \in \{1,\ldots,k\}\,.$$

We assume that these constraints are linearly independent for any $\boldsymbol{q}$. Equivalently, for each $\boldsymbol{q}$, there exists $m = n-k$ linearly independent vectors $f_1(\boldsymbol{q}),\ldots,f_m(\boldsymbol{q})$ such that the above constraints are equivalent to

$$\exists (u_1,\ldots,u_m) \in \mathbb{R}^m\,, \quad \dot{\boldsymbol{q}} = \sum_{i=1}^{m} u_i f_i(\boldsymbol{q})\,. \tag{47.1}$$

Let us note that the choice of vectors $f_i(\boldsymbol{q})$ is not unique. Fortunately, all the following developments are valid whatever choice we make. Moreover, if the linear constraints are smooth, vector fields $f_1,\ldots,f_m$ can be chosen smooth with respect to $\boldsymbol{q}$. We assume this condition from now on.

Let us define by $\mathcal{U}$ a compact subset of $\mathbb{R}^m$. We denote by Σ the control system defined by (47.1), with $(u_1,\ldots,u_m) \in \mathcal{U}$.

Definition 47.1

A local and small-time controllability:

1. Σ is *locally controllable about configuration* $\boldsymbol{q}$ iff the set of configurations reachable from $\boldsymbol{q}$ by an admissible trajectory contains a neighborhood of $\boldsymbol{q}$.
2. Σ is *small-time controllable about configuration* $\boldsymbol{q}$ iff the set of configurations reachable from $\boldsymbol{q}$ by an admissible trajectory in time less than T contains a neighborhood of $\boldsymbol{q}$ for any T.

$f_1,\ldots,f_m$ are called *control vector fields* of Σ. A system's small-time controllable about each configuration is said to be *small-time controllable*.

47.2.2 Controllability

Checking the controllability properties of a system requires the analysis of the control Lie algebra associated with the system. Let us illustrate in an informal way what the Lie bracket of two vector fields is. Consider two basic motions: *go along a straight line* and *turn on the spot*, supported by vector fields denoted by f and g, respectively. Now consider the following combination: go forward during a time t, turn clockwise during the same time t, go backward during the same time t and then turn counterclockwise during the time t. The system reaches a configuration that is not the starting one. Of course when t tends to 0 the goal configuration is very close to the starting one. The direction indicated by such goal configurations when t tends to 0 correspond to a new vector field, which is the Lie bracket of f and g. In a more mathematical formulation, the Lie

bracket $[f, g]$ of two vector fields f and g is defined as being the vector field $\partial f.g - \partial g.f$. The k-th coordinate of $[f, g]$ is

$$[f, g][k] = \sum_{i=1}^{n} \left(g[i] \frac{\partial}{\partial x_i} f[k] - f[i] \frac{\partial}{\partial x_i} g[k] \right) .$$

The following theorem [47.10] gives a powerful result for symmetric systems (a system is said to be symmetric when $\mathcal{U}$ is symmetric with respect to the origin).

Theorem 47.1
A symmetric system is small-time controllable about configuration $\boldsymbol{q}$ iff the rank of the vector space spanned by the family of vector fields f_i together with all their brackets is n at $\boldsymbol{q}$.

Checking the Lie algebra rank condition (LARC) on a control system consists in trying to build a basis of the tangent space from a basis (e.g., a P. Hall family) of the free Lie algebra spanned by the control vector fields. An algorithm is proposed in [47.11, 12].

47.2.3 Example: The Differentially-Driven Mobile Robot

To illustrate the notions developed in this section, we consider the differentially-driven mobile robot displayed in Fig. 47.1. The configuration space of this robot is $\mathbb{R}^2 \times \mathrm{S}^1$, and a configuration can be represented by $\boldsymbol{q} = (x, y, \theta)$ where (x, y) is the position in the horizontal plane of the center of the wheel axis of the robot and θ the orientation with respect to the x-axis. The rolling without slipping kinematic constraint

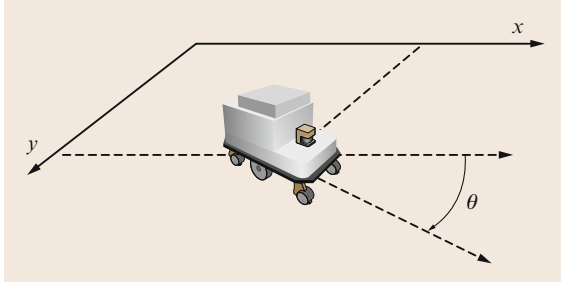


Fig. 47.1 A differentially-driven mobile robot is subject to one linear kinematic constraint, due to the rolling without slipping constraint of the wheel axis

$$-\dot{x} \sin\theta + \dot{y} \cos\theta = 0$$

is linear with respect to the velocity vector $(\dot{x}, \dot{y}, \dot{\theta})$. Therefore, the subspace of admissible velocities is spanned by two vector fields, for instance,

$$f_1(\boldsymbol{q}) = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} \quad \text{and} \quad f_2(\boldsymbol{q}) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} . \qquad (47.2)$$

The Lie bracket of these two vector fields is

$$f_3(\boldsymbol{q}) = \begin{pmatrix} \sin\theta \\ -\cos\theta \\ 0 \end{pmatrix} .$$

This implies that about any configuration $\boldsymbol{q}$, the rank of the vector space spanned by $f_1(\boldsymbol{q}), f_2(\boldsymbol{q}), f_3(\boldsymbol{q})$ is 3 and, therefore, that the differentially-driven mobile robot is small-time controllable.

47.3 Motion Planning and Small-Time Controllability

Motion planning raises two problems: the first one addresses the *existence* of a collision-free admissible path; this is the decision problem. The second one addresses the *computation* of such a path; this is the complete problem.

47.3.1 The Decision Problem

From now on, we assume that the set of collision-free configurations is an open subset. This implies that contact configurations are assumed in collision.

Theorem 47.2
The existence of a collision-free admissible path for a symmetric small-time controllable mobile robot between two configurations is equivalent to the existence of a collision-free (not necessarily admissible) path between these configurations.

Proof: Let us consider a not necessarily admissible collision-free path between two configurations $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ as a continuous mapping Γ from interval $[0, 1]$ into the configuration space CS such that:

1. $\Gamma(0) = \boldsymbol{q}_1$, $\Gamma(1) = \boldsymbol{q}_2$,
2. for any $t \in [0, 1]$, $\Gamma(t)$ is collision free.

Point 2 implies that for any t there exists a neighborhood $U(t)$ of $\Gamma(t)$ included in the collision-free subset of the configuration space.

Let us denote by $\varepsilon(t)$ the bigger lower bound of the time to collision of all the trajectories starting from

Avidbots Ex1016
Page 1207 of 2154

$\Gamma(t)$. As the control vector $(u_1, \ldots, u_m)$ remains in the compact set $\mathcal{U}$, $\varepsilon(t) > 0$.

As the system is small-time controllable about $\Gamma(t)$, the set reachable from $\Gamma(t)$ in time less than $\varepsilon(t)$ is a neighborhood of $\Gamma(t)$, which we denote by $V(t)$.

The collection $\{V(t), t \in [0,1]\}$ is an open covering of the compact set $\{\Gamma(t), t \in [0,1]\}$. Therefore, we can extract a finite covering, $\{V(t_1), \ldots, V(t_l)\}$, where $t_1 = 0 < t_2 < \ldots < t_{l-1} < t_l = 1$ such that for any i between 1 and $l-1$, $V(t_i) \cap V(t_{i+1}) \neq \emptyset$. For each i between 1 and $l-1$, we choose one configuration $\boldsymbol{r}_i$ in $V(t_i) \cap V(t_{i+1})$. As the system is symmetric, there exists an admissible collision-free path between $\boldsymbol{q}(t_i)$ and $\boldsymbol{r}_i$ and between $\boldsymbol{r}_i$ and $\boldsymbol{q}(t_{i+1})$. The concatenation of these paths is a collision-free admissible path between $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$. ■

47.3.2 The Complete Problem

In the former section, we have established that the decision problem, i. e., determining whether there exists a collision-free admissible path between two configurations, is equivalent to determining whether the configurations lie in the same connected component of the collision-free configuration space. In this section we present the tools necessary to solve the complete problem. These tools blend ideas from the classical motion planning problem addressed in Chap. 7 and from open loop control theory, but require specific developments that we are going to present in the next section. Two main approaches have been devised in order to plan admissible collision-free motions for nonholonomic systems. The first one proposed by [47.13] exploits the idea of the proof of Theorem 47.2 by recursively approximating a not necessarily admissible collision-free path by a sequence of feasible paths. The second approach replaces the local method of probabilistic roadmap method (PRM) algorithms (Chap. 7) by a local steering method that connects configuration pairs by admissible paths (VIDEO 707).

Both approaches use a steering method. Before briefly describing them, we give the definition of a local steering method.

Definition 47.2

A local steering method for system Σ is a mapping

$$S_{\text{loc}} : CS \times CS \to C^1_{\text{pw}}([0,1], CS)$$
$$(\boldsymbol{q}_1, \boldsymbol{q}_2) \mapsto S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$$

5 where $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$ is a piecewise continuously differentiable curve in CS satisfying the following properties:

1. $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$ satisfies the kinematic constraints associated to Σ,
2. $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$ connects $\boldsymbol{q}_1$ to $\boldsymbol{q}_2$: $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)(0) = \boldsymbol{q}_1$, $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)(1) = \boldsymbol{q}_2$.

Approximation of a Not Necessarily Admissible Path

A not necessarily admissible collision-free path $\Gamma(t), t \in [0,1]$ connecting two configurations and a local steering method S_{loc} being given, the approximation algorithm proceeds recursively, by calling function `approximation` defined by Algorithm 47.1 with input Γ, 0 and 1.

Algorithm 47.1

`approximation` function: inputs are a path Γ and two abscissas $t1$ and $t2$ along this path

if $S_{\text{loc}}(\Gamma(t1), \Gamma(t2))$ collision free **then**
  return $S_{\text{loc}}(\Gamma(t1), \Gamma(t2))$
else
  return concat(`approximation`(Γ,t1,(t1+t2)/2), `approximation`(Γ,(t1+t2)/2,t2))
end if

Sampling-Based Roadmap Methods

Most sampling-based roadmap methods as described in Chap. 6 can be adapted to nonholonomic systems by replacing the connection method between pairs of configurations by a local steering method. This strategy is rather efficient for PRM algorithms. For the rapidly-exploring random tree (RRT) method, the efficiency strongly depends on the metric used to choose the nearest neighbor. The distance function between two configurations needs to account for the length of the path returned by the local steering method to connect these configurations [47.14].

47.4 Local Steering Methods and Small-Time Controllability

The approximation algorithm described in the former section is recursive and raises the completeness question: does the algorithm finish in finite time or may it fail to find a solution?

A sufficient condition for the approximation algorithm to find a solution in a finite number of iterations is that the local steering method accounts for small-time controllability.

Definition 47.3

A local steering method S_{loc} accounts for the small-time controllability of system Σ iff

For any $\boldsymbol{q} \in CS$, for any neighborhood U of $\boldsymbol{q}$, there exists a neighborhood V of $\boldsymbol{q}$ such that for any $\boldsymbol{r} \in V$, $S_{\text{loc}}(\boldsymbol{q}, \boldsymbol{r})([0, 1]) \subset U$.

In other words, a local steering method accounts for the small-time controllability of a system if it produces paths getting closer to the configurations it connects when these configurations get closer to each other.

This property is also sufficient for probabilistic completeness of roadmap sampling-based methods.

47.4.1 Local Steering Methods Accounting for Small-Time Controllability

Constructing a local steering method that accounts for small-time controllability is a difficult task that has been achieved only for a few classes of systems. Most mobile robots studied in the domain of motion planning are wheeled mobile robots towing trailers or not.

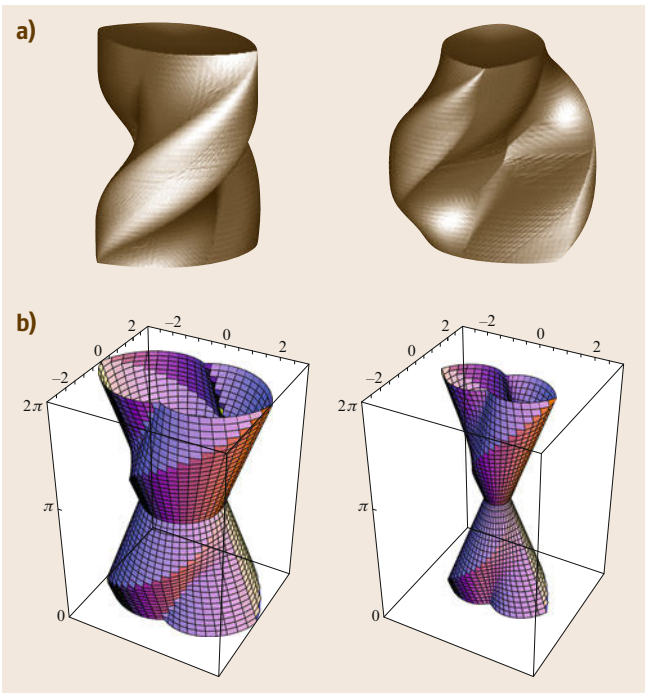


Fig. 47.2 **(a)** Two perspective views of an RS (Reeds and Shepp) ball: set of configurations reachable by a path of length less than a given distance for the Reeds and Shepp car. **(b)** Perspective views of two DD (differentially driven) balls: set of configurations reachable by a path of a length less than a given distance for the differentially-driven robot. Orientation θ is represented on z-axis

Steering Using Optimal Control

The simplest system, namely the differentially-driven robot presented in Sect. 47.2.3 with bounded velocities or the so-called *Reeds* and *Shepp* [47.15] car with bounded curvature have the same control vector fields (47.2). The difference lies in the domain of the control variables:

- $-1 \leq u_1 \leq 1, |u_2| \leq |u_1|$ for RS car,
- $|u_1| + b|u_2| \leq 1$ for the differentially-driven robot,

where b is half of the distance between the right and left wheels. For these systems, a local steering method accounting for small-time controllability can be constructed using optimal control theory. For any admissible path defined over an interval I of one of these systems, we define a length as follows

$$\int_I |u_1| \quad \text{for the RS car ,}$$

$$\int_I |u_1| + b|u_2| \quad \text{for the differentially-driven robot .}$$

The length of the shortest path between two configurations in both cases defines a metric over the configuration space. The synthesis of the shortest paths, i. e., the determination of the shortest path between any pair of configurations was achieved by [47.16] for the RS car and later by [47.17] for the differentially-driven robot. Figure 47.2 shows a representation of the balls corresponding to these metrics.

Optimal control naturally defines a local steering method that associates a shortest path between these configurations to any pair of configurations. Let us note that the shortest path is unique between most pairs of configurations. A general result states that the collection of balls of radius $r > 0$ centered about $\boldsymbol{q}$ induced by nonholonomic metrics constitutes an increasing collection of neighborhoods of $\boldsymbol{q}$, the intersection of which is $\{\boldsymbol{q}\}$. This property directly implies that local steering methods based on shortest paths account for small-time controllability.

The main advantage of optimal control is that it provides both a local steering method and a distance metric consistent with the steering method. This makes the steering methods well suited for path planning algorithms designed for holonomic systems and using a distance function, such as RRT (Chap. 7), for instance.

Unfortunately, the synthesis of the shortest paths has been realized only for the two simple systems described in this section. For more complex systems, the problem remains open.

The main drawback of the shortest path-based steering methods described in this section is that input

functions are not continuous. This requires an additional step to compute a time-parameterization of the paths before motion execution. Along this time-parameterization, input discontinuities force the robot to stop. For instance, to follow two successive arcs of circles of opposite curvature a mobile robot needs to stop between the arcs of circle in order to ensure continuity of the linear and angular velocities u_1 and u_2.

Steering Chained-Form Systems

Some classes of systems can be put into the form called the *chained-form* system by a change of variable

$$\dot{z}_1 = u_1 \,, \tag{47.3}$$

$$\dot{z}_2 = u_2 \,, \tag{47.4}$$

$$\dot{z}_3 = z_2 u_1 \,, \tag{47.5}$$

$$\vdots \quad \vdots$$

$$\dot{z}_n = z_{n-1} u_1 \,. \tag{47.6}$$

Let us consider the following inputs [47.18]

$$\begin{cases} u_1(t) &= a_0 + a_1 \sin \omega t \,, \\ u_2(t) &= b_0 + b_1 \cos \omega t \\ &\quad + \ldots + b_{n-2} \cos(n-2)\omega t \,. \end{cases} \tag{47.7}$$

Let $Z^{\text{start}} \in \mathbb{R}^n$ be a starting configuration. Each $z_i(1)$ can be computed from the coordinates of Z^{start} and parameters $(a_0, a_1, b_0, b_1, \ldots, b_{n-2})$. For a given $a_1 \neq 0$ and a given configuration Z^{start}, the mapping from $(a_0, b_0, b_1, b_2, b_3)$ to $Z(1)$ is a C^1-diffeomorphism at the origin; the system is then invertible. For n smaller or equal to 5, parameters $(a_0, b_0, b_1, \ldots, b_{n-2})$ can be analytically computed from the coordinates of the two configurations Z^{start} and Z^{goal}. The corresponding sinusoidal inputs steer the system from Z^{start} to Z^{goal}. The shape of the path only depends on parameter a_1. Each value of a_1 thus defines a local steering method denoted by $S^{a_1}_{\sin}$. None of these steering methods account for small-time controllability since for any $Z \in \mathbb{R}^n$, $S^{a_1}_{\sin}(Z, Z)([0, 1])$ is not reduced to $\{Z\}$. To construct a local steering method accounting for small-time controllability from the collection of $S^{a_1}_{\sin}$, we need to make a_1 depend on the configurations Z_1 and Z_2 that we want to connect

$$\lim_{Z^2 \to Z^1} a_1(Z^1, Z^2) = 0 \,,$$

$$\lim_{Z^2 \to Z^1} a_0\left[Z^1, Z^2, a_1(Z^1, Z^2)\right] = 0 \,,$$

$$\lim_{Z^2 \to Z^1} b_i\left[Z^1, Z^2, a_1(Z^1, Z^2)\right] = 0 \,.$$

Such a construction is achieved in [47.19].

Steering Feedback-Linearizable Systems

The concept of *feedback linearizability* (or differential flatness) was introduced by *Fliess* et al. [47.20, 21].

A system is said to be *feedback linearizable* if there exists an output (i. e., function of the state, input and input derivatives) called the *linearizing output*, such that the state and the input of the system is a function of the linearizing output and its derivatives. The dimension of the linearizing output is the same as the dimension of the input.

Let us illustrate this notion with a simple example. We consider a differentially-driven mobile robot towing a trailer hitched on top of the wheel axis of the robot, as displayed in Fig. 47.3. The tangent to the curve followed by the center of the wheel axis of the trailer gives the orientation of the trailer. From the orientation of the trailer along the curve, we can deduce the curve followed by the center of the robot. The tangent to the curve followed by the center of the robot gives the orientation of the robot. Thus the linearizing output of this system is the center of the wheel axis of the trailer. By differentiating the linearizing output twice, we can reconstruct the configuration of the system.

Feedback linearizability is very interesting for steering purposes. Indeed, the linearizing output is not subject to any kinematic constraints. Therefore, if we know the relation between the state and the linearizing output, planning an admissible path between two configurations simply consists in building a curve in $\mathbb{R}^m$,

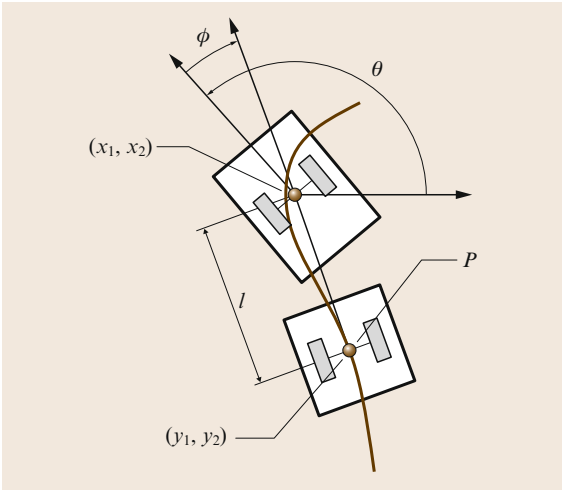


Fig. 47.3 A differentially-driven robot towing a trailer. Hitched on top of the wheel axis of the robot is a feedback-linearizable system. The linearizing output is the center of the wheel axis of the trailer. The configuration of the system can be reconstructed by differentiating the curve $\boldsymbol{y}(s)$, where s is an arc-length parameterization, followed by the linearizing output. The orientation of the trailer is given by $\tau = \arctan(\dot{y}_2/\dot{y}_1)$. The angle between the robot and the trailer is given by $\phi = -l \arctan(\mathrm{d}\tau/\mathrm{d}s)$. l is the length of the trailer connection

where m is the dimension of the input with differential constraints at both ends. This problem can be easily solved using, for instance, polynomials.

For two input driftless systems like Σ, the linearizing output only depends on state $\boldsymbol{q}$. The state $\boldsymbol{q}$ depends on the linearizing output through the parameterization invariant values, namely, the linearizing output $\boldsymbol{y}$, the orientation τ of the vector tangent to the curve followed by $\boldsymbol{y}$ and the successive derivatives of τ with respect to the curvilinear abscissa s.

Thus, the configuration of a two input feedback-linearizable driftless system of dimension n can be represented by a vector $(\boldsymbol{y}, \tau, \tau^1, \ldots, \tau^{n-3})$ representing the geometric properties of the curve followed by the linearizing output along an admissible path passing through the configuration.

Therefore, designing a local steering method for such a system is equivalent to associating to any pair of vectors $(\boldsymbol{y}_1, \tau_1, \tau_1^1, \ldots, \tau_1^{n-3})$, $(\boldsymbol{y}_2, \tau_2, \tau_2^1, \ldots, \tau_2^{n-3})$ a curve in the plane starting from $\boldsymbol{y}_1$ and ending at $\boldsymbol{y}_2$ with orientation of the tangent vector and successive derivatives with respect to s and equal to $\tau_1, \tau_1^1, \ldots, \tau_1^{n-3}$ at the beginning and to $\tau_2, \tau_2^1, \ldots, \tau_2^{n-3}$ at the end. This exercise is relatively easy using polynomials and transforming the boundary conditions into linear equations over the coefficients of the polynomials. However, taking into account small-time controllability is a little bit more tricky. Reference [47.22] proposes a flatness-based steering method built on convex combinations of canonical curves.

47.4.2 Equivalence Between Chained-Formed and Feedback-Linearizable Systems

In the previous section, we proposed methods to steer feedback-linearizable control systems or systems that can be put into chained-form. We now give a necessary and sufficient condition for feedback-linearizability.

Feedback-Linearizability: A Necessary and Sufficient Condition

In [47.23], *Rouchon* gives conditions to check whether or not a system is feedback-linearizable. For two-input driftless systems a necessary and sufficient condition is the following: let us define as Δ^k, $k > 0$ the collection of distributions (i. e., set of vector fields) iteratively defined by: $\Delta_0 = \text{span}\{f_1, f_2\}$, $\Delta_1 = \text{span}\{f_1, f_2, [f_1, f_2]\}$ and $\Delta_{i+1} = \Delta_0 + [\Delta_i, \Delta_i]$ with $[\Delta_i, \Delta_i] = \text{span}\{[f, g]\,,\, f \in \Delta_i,\, g \in \Delta_i\}$. A system with two-dimensional input is feedback-linearizable iff $\text{rank}(\Delta_i) = 2 + i$.

Example: The Chained-Form System

Let us consider the chained-form system defined by (47.3)–(47.6). The control vector fields of this system are

$$f_1 = (1, 0, z_2, \ldots, z_{n-1})\,,$$
$$f_2 = (0, 1, \ldots, 0)\,,$$

$\text{rank}\Delta_0 = 2$. If we compute $f_3 = [f_1, f_2] = (0, 0, 1, 0, \ldots, 0)$, we note that $\text{rank}\Delta_1 = 3$. By computing $f_i = [f_1, f_{i-1}]$ for i up to n, we find a sequence $f_i = (0, \ldots, 0, 1, 0, \ldots, 0)$, where 1 is at position i. Therefore, $\text{rank}\Delta_i = 2 + i$ for i up to $n-2$ and the chained-form system is feedback linearizable. This conclusion could have been drawn in a more straightforward way by noticing that the state can be reconstructed from (z_1, z_2) and its derivatives. (z_1, z_2) is thus the linearizing output of the chained-form system.

47.5 Robots and Trailers

Robotics systems studied in motion planning are mainly those composed of a mobile robot alone or towing one or several trailers. The input of these systems is two dimensional.

47.5.1 Differentially-Driven Mobile Robots

The simplest mobile robot, namely the differentially-driven mobile robot displayed in Fig. 47.1, is obviously feedback-linearizable. the trajectory of the center of the wheel axis (the linearizing output) of the robot gives the orientation of the robot. It can thus be steered using a flatness-based local steering method.

47.5.2 Differentially-Driven Mobile Robots Towing One Trailer

The differentially-driven mobile robot towing a trailer hitched on top of the wheel axis of the robot displayed in Fig. 47.3 is feedback linearizable and the linearizing output is the center of the wheel axis of the trailer. The differentially-driven mobile robot towing a trailer with hitched kingpin (Fig. 47.4) is also feedback lineariz-

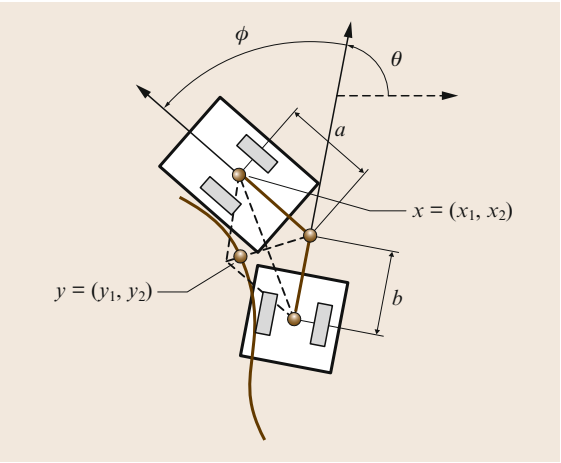


Fig. 47.4 A differentially-driven mobile robot towing a trailer with hitched kingpin is feedback linearizable

able [47.24] but the relation between linearizing output and configuration variables is more intricate

$$\begin{cases} y_1 = x_1 - b\cos\theta + L(\phi)\dfrac{b\sin\theta + a\sin(\theta+\phi)}{\sqrt{a^2+b^2+2ab\cos\phi}}\,, \\ y_2 = x_2 - b\sin\theta - L(\phi)\dfrac{a\cos(\theta+\phi) + b\cos\theta}{\sqrt{a^2+b^2+2ab\cos\phi}}\,, \end{cases}$$

where L is defined by the following elliptic integral

$$L(\phi) = ab\int_0^{\phi} \frac{\cos\sigma}{\sqrt{a^2+b^2+2ab\cos(\sigma)}}\mathrm{d}\sigma\,,$$

which is the linearizing output of the system. These relations together with the following ones

$$\tan\tau = \frac{a\sin(\theta+\phi) + b\sin\theta}{b\cos\theta + a\cos(\theta+\phi)}\,,$$

$$\kappa = \frac{\sin(\phi)}{\cos\phi\sqrt{a^2+b^2+2ab\cos\phi} + L(\phi)\sin(\phi)}\,,$$

make it possible to reconstruct the configuration variables from the linearizing output and its two first derivatives.

47.5.3 Car-Like Mobile Robots

A car-like mobile robot is constituted of a fixed rear wheel axis and of two steerable front wheels, the axes of which intersect at the center of curvature (Fig. 47.5). A car-like mobile robot is kinematically equivalent to a differentially-driven mobile robot towing a trailer hitched on top of the wheel axis of the robot (Fig. 47.3): the virtual front wheel corresponds to the differentially-driven robot, while the body of the car corresponds to the trailer.

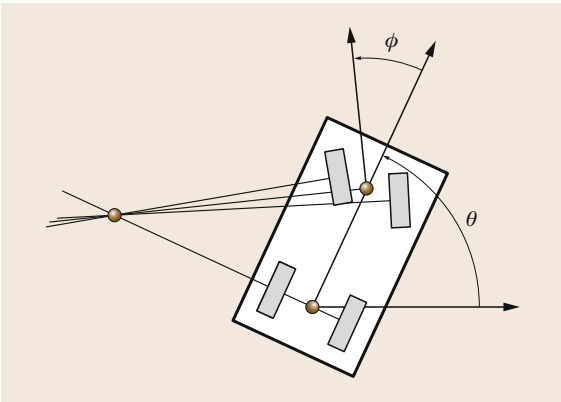


Fig. 47.5 Car-like mobile robot. The axes of the front wheels intersect at the center of curvature. The steering angle ϕ is the angle between the longitudinal axis of the car and a virtual front wheel in the middle of the two front wheels

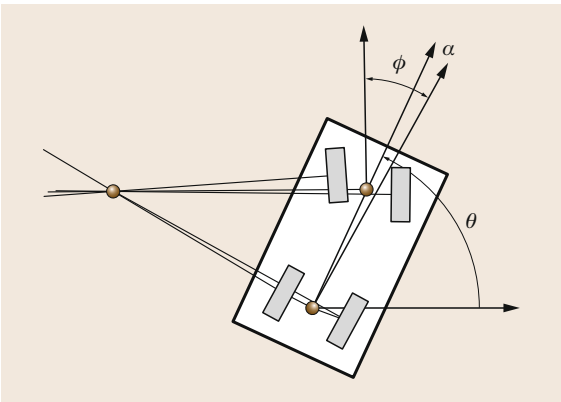


Fig. 47.6 Bi-steerable robot. The front and rear wheels are steerable and there is a relation between the rear steering angle and the front steering angle: $\alpha = f(\phi)$

47.5.4 Bi-Steerable Mobile Robots

The bi-steerable mobile robot (Fig. 47.6) is a car with front and rear steerable wheels and with a relation between the front and rear steering angles. This system has been proved to be feedback linearizable in [47.25]. As for the mobile robot towing a trailer with hitched kingpin, the linearizing output is a moving point in the robot reference frame.

47.5.5 Differentially-Driven Mobile Robots Towing Trailers

Let us consider the differentially-driven mobile robot towing a trailer connected on top of the wheel axis of the robot, and let us add an arbitrary number of trailers,

Avidbots Ex1016
Page 1212 of 2154

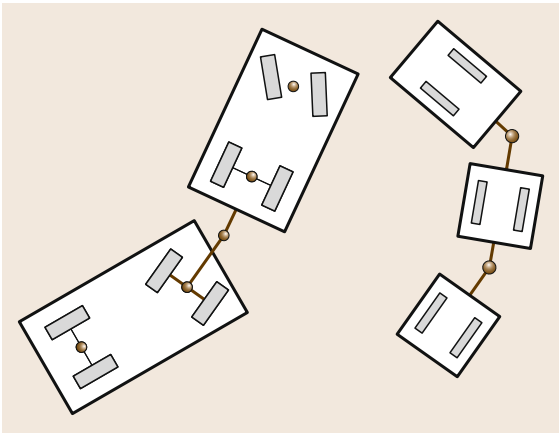


Fig. 47.7 A truck towing a towbar trailer with hitched kingpin and a differentially-driven mobile robot towing two trailers with hitched kingpin: two open problems for exact path planning for nonholonomic systems

each one connected on top of the wheel axis of the previous one. By differentiating once the curve followed by the center of the last trailer, we get the orientation of the last trailer (this orientation coincides with the orientation of the tangent to the curve). If we know the orientation and the position of the center of the last trailer along the path, we can reconstruct the curve followed by the center of the previous trailer. Repeating this reasoning, we can reconstruct the trajectory of the whole system by differentiating a sufficient number of times. The system is therefore feedback linearizable and the linearizing output is the center of the last trailer.

Combining the above reasoning with the systems reviewed in this section, we can build hybrid feedback-linearizable trailer systems. For instance, a differentially-driven mobile robot towing an arbitrary number n of trailers each one connected on top of the wheel axis of the previous trailer, except the last trailer with hitched kingpin is feedback linearizable. Simply consider the two last trailers as a feedback-linearizable system composed of a mobile robot with one trailer as in Fig. 47.4. The linearizing output of this system enables us to reconstruct the trajectory of the two last trailers. The center of trailer $n-1$ is a linearizing output for the mobile robot towing the $n-1$ last trailers.

47.5.6 Open Problems

Finally, all the systems for which we are able to plan exact motions between arbitrary pairs of configurations are included in the large class of feedback-linearizable systems. We have indeed seen that chained-form systems are also part of this class. For other systems, no exact solution has been proposed up to now. For instance, neither of the systems displayed in Fig. 47.7 is feedback linearizable. They do not satisfy the necessary condition of Sect. 47.4.2.

47.6 Approximate Methods

To deal with nonholonomic systems not belonging to any class of systems for which exact solutions exist, numerical approximate solutions have been developed. We review some of these methods in this section.

47.6.1 Forward Dynamic Programming

In [47.26] *Barraquand* and *Latombe* propose a dynamic programming approach to nonholonomic path planning. Admissible paths are generated by a sequence of constant input values, applied over a fixed interval of time δt. Starting from the initial configuration the search generates a tree: the children of a given configuration $\boldsymbol{q}$ are obtained by setting the input to a constant value and integrating the differential system over δt. The configuration space is discretized into an array of cells of equal size (i. e., hyper-parallelepipeds). A child $\boldsymbol{q}'$ of a configuration $\boldsymbol{q}$ is inserted in the search tree if and only if the computed path from $\boldsymbol{q}$ to $\boldsymbol{q}'$ is collision free and $\boldsymbol{q}'$ does not belong to a cell containing an already generated configuration. The algorithm stops when it generates a configuration belonging to the same cell as the goal (i. e., it does not necessarily reach the goal exactly).

The algorithm has been proved to be asymptotically complete with respect to both δt and the size of the cells. As a brute force method, it remains quite time consuming in practice. Its main interest is that the search is based on Dijkstra's algorithm, which allows taking into account optimality criteria such as the path length or the number of reversals. Asymptotical optimality to generate the minimum of reversals is proved for the car-like robot alone.

47.6.2 Discretization of the Input Space

In [47.27] *Divelbiss* and *Wen* propose a method to produce an admissible collision-free path for a nonholonomic mobile robot in the presence of obstacles. They restrict the set of input functions over the subspace spanned by Fourier basis over interval $[0, 1]$. An in-

put function is thus represented by a finite-dimensional vector λ. Reaching a goal configuration thus becomes a nonlinear system of equations, the unknowns of which are the coordinates λ_i's of λ. The authors use the Newton–Raphson method to find a solution. Obstacles are defined by inequality constraints over the configuration space. The path is discretized into N samples. The noncollision constraint expressed at these sample points yields the inequality constraint over vector λ. These inequality constraints are turned into equality constraints through function g defined as

$$g(c) = \begin{cases} (1-\mathrm{e}^c)^2 & \text{if } c > 0\,, \\ 0 & \text{if } c \le 0\,. \end{cases}$$

Reaching a goal configuration while avoiding obstacles thus becomes a nonlinear system of equations over vector λ again solved using the Newton–Raphson method. The method is rather efficient for short maneuvers. The main difficulty is to tune the order of the Fourier expansion. Long motions in cluttered environments require a higher order, while motion in empty space can be solved with a low order. The authors do not mention the problem of numerical instability in the integration of the dynamic system.

47.6.3 Input-Based Rapidly Exploring Random Trees

The RRT algorithms described in Chap. 7 can be used without a local steering method to plan paths for nonholonomic systems. New nodes can be generated from existing nodes by applying random input functions over an interval of time [47.28]. The main difficulty consists in finding a distance function that really accounts for the distance the system needs to travel to go from one configuration to another. Moreover, the goal is never exactly reached. This latter drawback can be overcome by post-processing the path returned by RRT using a path deformation method, as described in [47.29].

47.7 From Motion Planning to Obstacle Avoidance

Up to now we have described motion planning techniques. Their objective is to compute a collision-free trajectory to the target configuration that complies with the vehicle constraints. They assume a perfect model of the robot and scenario. The advantage of these techniques is that they provide complete and global solutions of the problem. Nevertheless, when the surroundings are unknown and unpredictable, these techniques fail.

A complementary way to face the motion problem is obstacle avoidance. The objective is to move a vehicle towards a target location free of collisions with the obstacles collected by the sensors during motion execution. The advantage of reactive obstacle avoidance is to compute motion by introducing the sensor information within the control loop, used to adapt the motion to any contingency incompatible with initial plans.

The main cost of considering the reality of the world during execution is locality. In this instance, if global reasoning is required, a trap situation could occur. Despite this limitation, obstacle avoidance techniques are mandatory to deal with mobility problems in unknown and evolving surroundings.

Notice that methods have been developed to combine both the global point of view of motion planning and the local point of view of obstacle avoidance. How can we consider robot perception at the planning level? This is so-called sensor-based motion planning. Several variants exists, such as the Bug algorithms initially introduced in [47.30]. However, non of them consider the practical context of nonholonomic mobile robots.

47.8 Definition of Obstacle Avoidance

Let $\mathcal{A}$ be the robot (a rigid object) moving in the workspace $\mathcal{W}$, whose configuration space is CS. Let $\boldsymbol{q}$ be a configuration, $\boldsymbol{q}_t$ this configuration in time t, $\mathcal{A}(\boldsymbol{q}_t) \in \mathcal{W}$ the space occupied by the robot in this configuration. In the vehicle there is a sensor, which in $\boldsymbol{q}_t$ measures a portion of the space $S(\boldsymbol{q}_t) \subset \mathcal{W}$ identifying a set of obstacles $\mathcal{O}(\boldsymbol{q}_t) \subset \mathcal{W}$.

Let $\boldsymbol{u}$ be a constant control vector and $\boldsymbol{u}(\boldsymbol{q}_t)$ this control vector applied in $\boldsymbol{q}_t$ during time δt. Given $\boldsymbol{u}(\boldsymbol{q}_t)$, the vehicle describes a trajectory $\boldsymbol{q}_{t+\delta t} = f(\boldsymbol{u}, \boldsymbol{q}_t, \delta t)$, with $\delta t \ge 0$. Let $\mathcal{Q}_{t,T}$ be the set of configurations of the trajectory followed from $\boldsymbol{q}_t$ with $\delta t \in [0, T]$, a given time interval. $T > 0$ is called the *sampling period*. Let $F : CS \times CS \to \mathbb{R}^+$ be a function that evaluates the progress of one configuration to another.

47.8.1 Obstacle Avoidance Problem

Let $\boldsymbol{q}_{\text{target}}$ be a target configuration. Then, in time t_i the robot $\mathcal{A}$ is in $\boldsymbol{q}_{t_i}$, where a sensor measurement

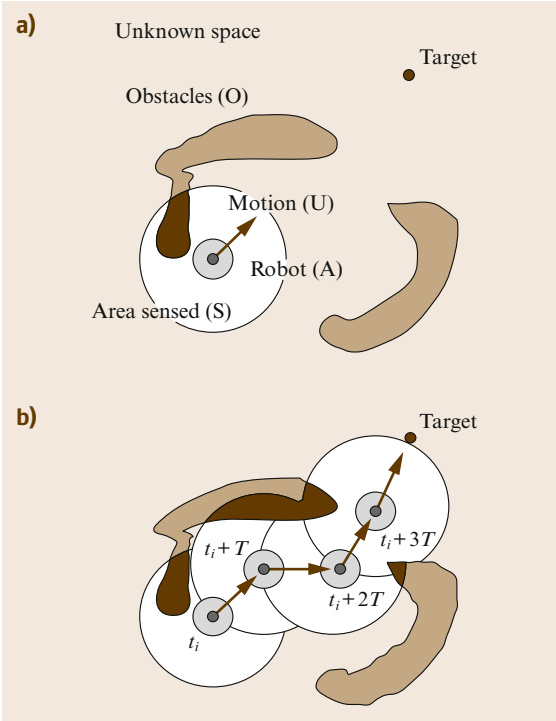


Fig. 47.8 **(a)** Obstacle avoidance problem consists of computing motion control that avoids collisions with the obstacles gathered by the sensors whilst driving the robot towards the target location. **(b)** The result of applying this technique at each time is a sequence of motions that drive the vehicle free of collisions to the target

is obtained $S(\boldsymbol{q}_{t_i})$*, and thus an obstacle description* $\mathcal{O}(\boldsymbol{q}_{t_i})$.

The objective is to compute a motion control $\boldsymbol{u}_i$*, such that* (*i*) *the trajectory generated is free of collisions with the obstacles* $\mathcal{A}(\mathcal{Q}_{t_i,T}) \bigcap \mathcal{O}(\boldsymbol{q}_{t_i}) = \emptyset$*; and* (*ii*) *it makes the vehicle progress to the target location* $F(\boldsymbol{q}_{t_i}, \boldsymbol{q}_{\text{target}}) < F(\boldsymbol{q}_{t_i+T}, \boldsymbol{q}_{\text{target}})$.

The result of solving this problem at each sample time (Fig. 47.8a) is a sequence of motion controls $\{\boldsymbol{u}_1 \ldots \boldsymbol{u}_n\}$ computed in execution time that avoids the obstacles gathered by the sensors, while making the vehicle progress towards the target location in each configuration $\{\boldsymbol{q}_{t_1} \ldots \boldsymbol{q}_{\text{target}}\}$ (Fig. 47.8b). Notice that the motion problem is a global problem. The obstacle avoidance methods are local and iterative techniques to address this problem. The disadvantage of locality (trap situations) is counteracted with the advantage of introducing the sensory information within the control cycle (to take into account the reality of the world in execution).

There are at least three aspects that affect the development of an obstacle avoidance method: the avoidance technique, the type of robot-sensor and the type of scenario. These subjects correspond to the next three sections. First, we describe the obstacle avoidance techniques (Sect. 47.9). Second, we discuss the techniques to adapt a given obstacle avoidance method to work on a vehicle taking into account the shape, kinematics, and dynamics (Sect. 47.10). The sensory processing is detailed in Chaps. 5 and 25 of this book. Finally, the usage of an obstacle avoidance technique on a vehicle in a given scenario is highly dependent on the scenario nature (static or dynamic, unknown or known, structured or not, or its size, for example). Usually, this problem is associated with the integration of planning and obstacle avoidance (Sect. 47.11).

47.9 Obstacle Avoidance Techniques

We describe here a taxonomy of obstacle avoidance techniques and some representative methods. First there are two groups, methods that compute the motion in one step and methods that do it in more than one. Methods of one step directly reduce the sensor information to motion control. There are two types:

- The *heuristic* methods were the first techniques used to generate motion based on sensors. The majority of this work was derived in classic planning methods and will not be described here. See [47.1, 30–33] for some representative work.
- The methods of *physical analogies* assimilate the obstacle avoidance to a known physical problem. We discuss here the potential field methods [47.34, 35]. Other works are variants adapted to uncertain models [47.36] or that use other analogies [47.37–39].

Methods with more than one step compute some intermediate information, which is processed next to obtain the motion:

- The methods of *subset of controls* compute an intermediate set of motion controls, and next they choose one of them as a solution. There are two types: (i) methods that compute a subset of motion directions. We describe here the vector field histogram (VFH) [47.40] and the obstacle restriction method [47.41] (see [47.42] for three-dimensional (3-D) workspaces). Another method is the steer angle field approach [47.43]. (ii) Methods that com-

pute a subset of velocity controls. We describe here the dynamic window approach (DWA) [47.44] and velocity obstacles [47.45] (see [47.46] for moving obstacles). Another method based on similar principles but developed independently is the curvature velocity method [47.47].

- Finally, there are methods that compute some *high-level information* such as intermediate information, which is translated next in motion. We describe the basic versions of the nearness diagram navigation [47.48, 49]. The reader is directed to [47.50–52] for further improvements.

All the methods outlined here have advantages and disadvantages, depending on the navigation context, like uncertain worlds, motion at high speeds, motion in confined or troublesome spaces, etc. Unfortunately there is no metric available to quantitatively measure the performance of the methods. However, for an experimental comparison in terms of their intrinsic problems see [47.48].

47.9.1 Potential Field Methods

The potential field method (PFM) uses an analogy in which the robot is a particle that moves in the configuration space under the influence of a force field. While the target location exerts a force that attracts the particle $\boldsymbol{F}_{\text{att}}$, the obstacles exert repulsive forces $\boldsymbol{F}_{\text{rep}}$. At each time t_i, the motion is computed to follow the direction of the artificial force induced by the sum of both potentials $\boldsymbol{F}_{\text{tot}}(\boldsymbol{q}_{t_i}) = \boldsymbol{F}_{\text{att}}(\boldsymbol{q}_{t_i}) + \boldsymbol{F}_{\text{rep}}(\boldsymbol{q}_{t_i})$ (the most promising motion direction), Fig. 47.9.

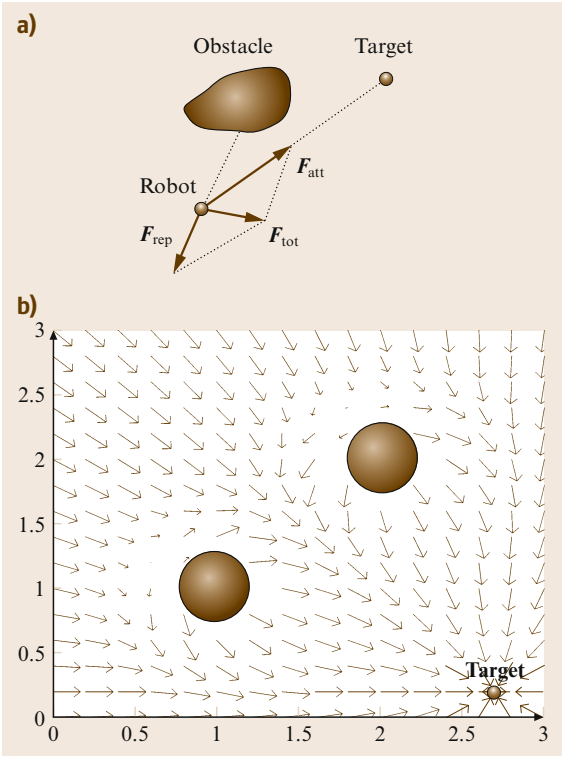


Fig. 47.9 **(a)** Computation of the motion direction with a potential field method. The target attracts the particle $\boldsymbol{F}_{\text{att}}$ while the obstacle exerts a repulsive force $\boldsymbol{F}_{\text{rep}}$. The resulting force $\boldsymbol{F}_{\text{tot}}$ is the most promising motion direction. **(b)** Motion directions computed in each point of the space with the classic method

Example

$$\boldsymbol{F}_{\text{att}}(\boldsymbol{q}_{t_i}) = K_{\text{att}}\boldsymbol{n}_{q_{\text{target}}} \tag{47.8}$$

$$\boldsymbol{F}_{\text{rep}}(\boldsymbol{q}_{t_i}) = \begin{cases} K_{\text{rep}} \sum_j \left(\dfrac{1}{d(\boldsymbol{q}_{t_i}, \boldsymbol{p}_j)} - \dfrac{1}{d_0} \right) \boldsymbol{n}_{p_j} & \text{if } d(\boldsymbol{q}_{t_i}, \boldsymbol{p}_j) < d_0 \,, \\ 0 & \text{otherwise} \end{cases} \tag{47.9}$$

where K_{att} and K_{rep} are the constants of the forces, d_0 is the influence distance of the obstacles $\boldsymbol{p}_j$, $\boldsymbol{q}_{t_i}$ is the current vehicle configuration and $\boldsymbol{n}_{q_{\text{target}}}$ and $\boldsymbol{n}_{p_j}$ are the unitary vectors that point from $\boldsymbol{q}_{t_i}$ to the target and each obstacle $\boldsymbol{p}_j$. From $\boldsymbol{F}_{\text{tot}}(\boldsymbol{q}_{t_i})$, the control $\boldsymbol{u}_i$ can be obtained with a position or force control [47.53].

This is the classic version, where the potentials only depend on the current vehicle configuration. Complementarily the generalized potentials depend also on the instantaneous robot velocity and vehicle accelerations.

Example

$$\boldsymbol{F}_{\text{rep}}(\boldsymbol{q}_{t_i}) = \begin{cases} K_{\text{rep}} \sum_j \left(\dfrac{a\dot{\boldsymbol{q}}_{t_i}}{[2ad(\boldsymbol{q}_{t_i}, \boldsymbol{p}_j) - \dot{\boldsymbol{q}}_{t_i}^2]} \right) \boldsymbol{n}_{p_j} \cdot \boldsymbol{n}_{\dot{\boldsymbol{q}}_{t_i}} & \text{if } \dot{\boldsymbol{q}}_{t_i} > 0 \,, \\ 0 \quad \text{otherwise} \end{cases} \tag{47.10}$$

where $\dot{\boldsymbol{q}}_{t_i}$ is the current robot velocity, $\boldsymbol{n}_{\dot{\boldsymbol{q}}_{t_i}}$ the unitary vector pointing in the direction of the robot velocity, and a is the maximum vehicle acceleration. This expression arises when the repulsive potential is defined as the inverse of the difference between the estimated time until a collision takes place and the time needed to stop the robot until full backward acceleration. Notice how the repulsive only affect in the direction of motion of the vehicle, as opposed to the classic potential. For a comparison of the classic and generalized versions,

and the relation with the way to compute the motion controls next, see [47.53]. This method is widely used because it is easy to understand and due to its clear mathematic formalism.

47.9.2 Vector Field Histogram

The vector field histogram (VFH) solves the problem in two steps by computing a set of candidate motion directions and then selecting one of them.

Candidate Set of Directions

Firstly the space is divided into sectors from the robot location. The method uses a polar histogram H constructed around the robot, where each component represents the obstacle polar density in the corresponding sector. The function that maps the obstacle distribution in sector k on the corresponding component of the histogram $h^k(\boldsymbol{q}_{t_i})$ is

$$h^k(\boldsymbol{q}_{t_i}) = \int_{\Omega_k} P(\boldsymbol{p})^n . \left(1 - \frac{d(\boldsymbol{q}_{t_i}, \boldsymbol{p})}{d_{\max}}\right)^m \mathrm{d}\boldsymbol{p} \,. \qquad (47.11)$$

The dominion of integration is $\Omega_k = \{\boldsymbol{p} \in \mathcal{W} \backslash \boldsymbol{p} \in k \wedge d(\boldsymbol{q}_{t_i}, \boldsymbol{p}) < d_0\}$. The density $h^k(\boldsymbol{q}_{t_i})$ is proportional to the probability $P(\boldsymbol{r})$ that a point was occupied by an obstacle, and to a factor that increases when the distance to the point decreases.

Typically the resulting histogram has peaks (directions with high density of obstacles) and valleys (directions with low density). The set of candidate directions is the set of adjacent components with density lower than a given threshold, and closest to the component that contains the target direction. This set of components (sectors) is called the selected valley, and represents a set of candidate directions (Fig. 47.10).

Motion Computation

The objective of the next step is to choose a direction of this set. The strategy is to apply three heuristics that depend on the component that contains the target or on the size of the selected valley. The cases are checked in sequence:

1. Case 1: goal sector in the selected valley. Solution: $k_{\mathrm{sol}} = k_{\mathrm{target}}$, where k_{target} is the sector that contains the goal location.
2. Case 2: goal sector not in the selected valley and number of sectors of the valley greater than m. Solution: $k_{\mathrm{sol}} = k_i \pm m/2$, where m is a fixed number of sectors and k_i the sector of the valley closer to the k_{target}.

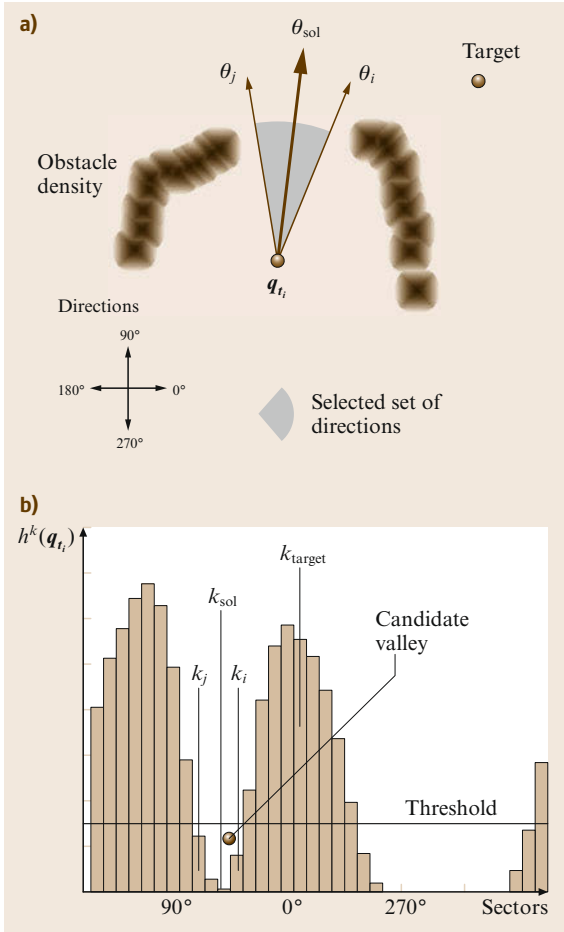


Fig.47.10a,b Computation of the motion direction θ_{sol} with VFH. **(a)** Robot and obstacle occupancy distribution. **(b)** The candidate valley is the set of adjacent components with values lower than the threshold. The navigation case is Case 3, since the sector of the target k_{target} is not in the valley and the number of sectors is lower than a fixed quantity m ($m = 8$, i. e., 45°). Thus the solution is $k_{\mathrm{sol}} = (k_i + k_j)/2$, whose bisector is θ_{sol} in **(a)**. The bisectors of k_i and k_j are θ_i and θ_j, respectively

3. Case 3: goal sector not in the selected valley and number of sectors of the valley lower or equal to m. Solution: $k_{\mathrm{sol}} = (k_i + k_j)/2$, where k_i and k_j are the extreme sectors of the valley.

The result is a component or sector k_{sol}, whose bisector is the direction solution θ_{sol}. The velocity v_{sol} is inversely proportional to the distance to the closest obstacle. The control is $\boldsymbol{u}_i = (v_{\mathrm{sol}}, \theta_{\mathrm{sol}})$.

VFH is a method formulated to work with probability obstacle distributions, and thus, it is well adapted to work with uncertain sensors as ultrasonic sonars.

Avidbots Ex1016
Page 1217 of 2154

47.9.3 The Obstacle Restriction Method

The obstacle restriction method (ORM) solves the problem in three steps, where the result of the two first steps is a set of candidate motion directions. The first step is to compute an instantaneous subgoal if necessary. The second one associates a motion constraint to each obstacle and joins them next to compute the set of desirable directions. The last step is an strategy to compute the motion given this set.

Instantaneous Target Selection

This step computes a subgoal when is better to direct the motion towards a given zone of the space (which ameliorates the situation to reach the goal latter), rather than directly towards the goal itself. The subgoals are located in between obstacles or in the edge of an obstacle (Fig. 47.11a). Next, the process is to check with a local algorithm whether the goal can be reached from the robot location. If it is not, then the closest reachable subgoal to the goal is selected. To check if a point can be reached, there is a local algorithm that computes the existence of a local path that joins two locations [47.48].

Let $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ be two locations of the space, R the robot radius, and L a list of obstacle points, where $\boldsymbol{p}_i$ is an obstacle of the list. Let $\mathcal{A}$ and $\mathcal{B}$ be the two semiplanes divided by the line that joins $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$. Then, if for all the points of L, $d(\boldsymbol{p}_j, \boldsymbol{p}_k) > 2R$ (with $\boldsymbol{p}_j \in \mathcal{A}$ and $\boldsymbol{p}_k \in \mathcal{B}$), there is a collision free path that joins both locations. If this condition is not satisfied, then there is no local path (although a global one could exist). The interesting result is when the result is positive, since the guarantee that one point can be reached from the other exists.

The result of the process is the goal or an instantaneous subgoal (from now this location is called the target location). Note that this process is general and can be used as a preprocess step by other methods to instantaneously validate the goal location or to compute an instantaneous subgoal to drive the vehicle.

Candidate Set of Directions

For each obstacle i a set of not desirable directions for motion $S^i_{n\mathrm{D}}$ is computed (motion constraint). This set is the union of two subsets S^i_1 and S^i_2. S^i_1 represents the side of the obstacle not suitable to do the avoidance and S^i_2 is an exclusion area around the obstacle (Fig. 47.11b). The motion constraint for the obstacle is the union of both sets $S^i_{n\mathrm{D}} = S^i_1 \cup S^i_2$. The set of desired directions of motion is the complementary $S_\mathrm{D} = \{[-\pi, \pi] \setminus S_{n\mathrm{D}}\}$, where $S_{n\mathrm{D}} = \cup_i S^i_{n\mathrm{D}}$.

Motion Computation

The final step is to select a direction of motion. There are three cases that depend on the set of desirable directions S_D and on the target direction θ_{target}. The cases are checked in sequence:

1. Case 1: $S_\mathrm{D} \neq \emptyset$ and $\theta_{\mathrm{target}} \in S_\mathrm{D}$. Solution: $\theta_{\mathrm{sol}} = \theta_{\mathrm{target}}$.
2. Case 2: $S_\mathrm{D} \neq \emptyset$ and $\theta_{\mathrm{target}} \notin S_\mathrm{D}$. Solution: $\theta_{\mathrm{sol}} = \theta_{\mathrm{lim}}$, where θ_{lim} is the direction of S_D closest to θ_{target}.
3. Case 3: $S_\mathrm{D} = \emptyset$. Solution: $\theta_{\mathrm{sol}} = (\phi^l_{\mathrm{lim}} + \phi^r_{\mathrm{lim}})/2$, where ϕ^l_{lim} and ϕ^r_{lim} are the directions of $S_{\mathrm{D_r}}$ and $S_{\mathrm{D_l}}$ closer to θ_{target} ($S_{\mathrm{D_r}}$ and $S_{\mathrm{D_l}}$ are the set of desirable

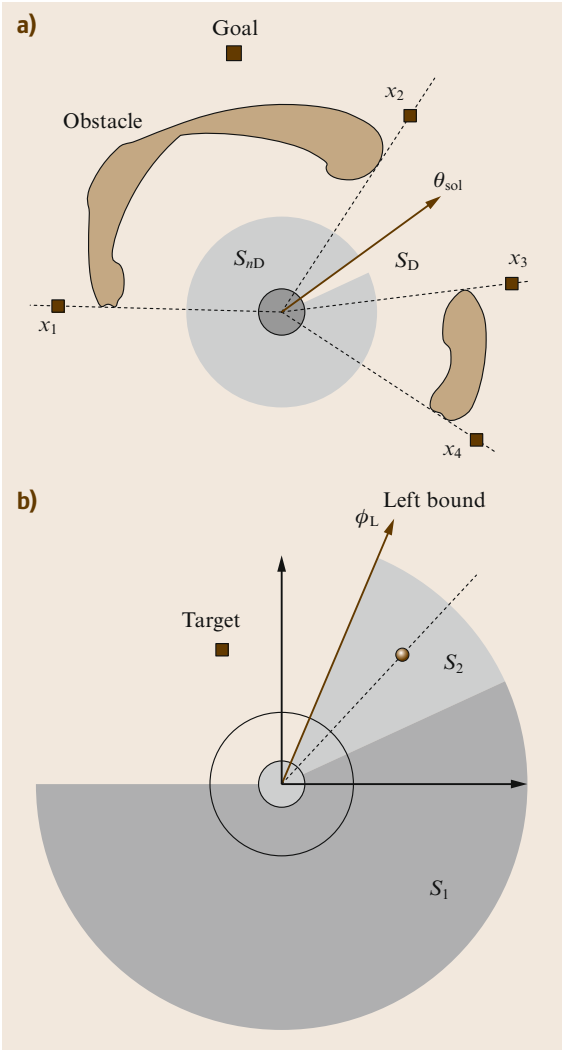


Fig. 47.11 **(a)** Distribution of the subgoals x_i. The selected instantaneous target location is x_2. The set of candidate directions is $S_{n\mathrm{D}}$ and the solution is θ_{sol} (the second case). **(b)** The two sets of not desirable directions S_1 and S_2 for a given obstacle

directions of obstacles on the left and right-hand sides of the target, respectively).

The result is a direction of motion solution θ_{sol}. The velocity v_{sol} is inversely proportional to the distance to the closest obstacle. The control is $\boldsymbol{u}_i = (v_{sol}, \theta_{sol})$. This is a geometric-based method based on cases. The advantage is that it has been demonstrated to address effective motion in confined spaces.

47.9.4 Dynamic Window Approach

The dynamic window approach (DWA) (VIDEO 712) is a method that solves the problem in two steps, by computing as intermediate information a subset of the control space $\mathcal{U}$. For simplicity, we consider a motion control such as translational and rotational velocity (v, w); $\mathcal{U}$ is defined by

$$\begin{aligned}\mathcal{U} = \{\, &(v, w) \in \mathbb{R}^2 \setminus \\ &v \in [-v_{max}, v_{max}] \wedge \\ &w \in [-w_{max}, w_{max}] \,\}\,. \end{aligned} \tag{47.12}$$

Set of Candidate Controls

The candidate set of controls $\mathcal{U}_R$ contains the controls: (i) within the maximum velocities of the vehicle $\mathcal{U}$, (ii) that generate safe trajectories $\mathcal{U}_A$, and (iii) that can be reached within a short period of time given the vehicle accelerations $\mathcal{U}_D$. The set $\mathcal{U}_A$ contains the admissible controls. These controls can be canceled before collision by applying the maximum deceleration (a_v, a_w)

$$\mathcal{U}_A = \left\{(v, w) \in \mathcal{U} | v \leq \sqrt{2 d_{obs} a_v} \wedge w \leq \sqrt{2 \theta_{obs} a_w}\right\}, \tag{47.13}$$

where d_{obst} and θ_{obst} are the distance to the obstacle and the orientation of the tangent to the trajectory in the obstacle. The set $\mathcal{U}_D$ contains the controls reachable in a short period

$$\begin{aligned}\mathcal{U}_D = \{\, &(v, w) \in \mathcal{U} \setminus \\ &v \in [v_o - a_v T, v_o + a_v T] \wedge \\ &w \in [w_o - a_w T, w_o + a_w T]\}\,, \end{aligned} \tag{47.14}$$

where $\dot{\boldsymbol{q}}_{t_i} = (v_o, w_o)$ is the current velocity.

The resulting subset of controls is (Fig. 47.12)

$$\mathcal{U}_R = \mathcal{U} \cap \mathcal{U}_A \cap \mathcal{U}_D\,. \tag{47.15}$$

Motion Computation

The next step is to select one control $\boldsymbol{u}_i \in \mathcal{U}_R$. The problem is set out as the maximization of an objective function

$$G(\boldsymbol{u}) = \alpha_1 \cdot \text{Goal}(\boldsymbol{u}) + \alpha_2 \cdot \text{Clearance}(\boldsymbol{u}) + \alpha_3 \cdot \text{Velocity}(\boldsymbol{u})\,. \tag{47.16}$$

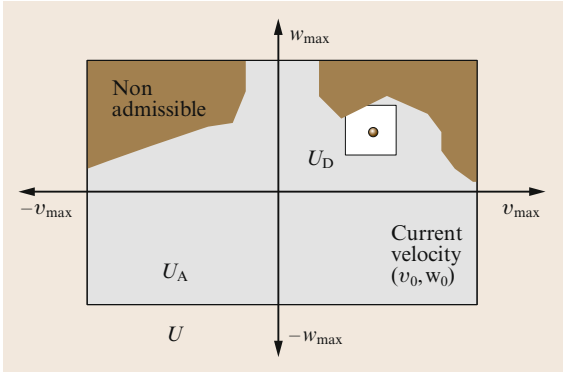


Fig. 47.12 Subset of controls $\mathcal{U}_R = \mathcal{U} \cap \mathcal{U}_A \cap \mathcal{U}_D$, where $\mathcal{U}$ contains the controls within the maximum velocities, $\mathcal{U}_A$ the admissible controls, and $\mathcal{U}_D$ the controls reachable in a short period of time

This function is a compromise among Goal$(\boldsymbol{u})$ that favors velocities that offer progress to the goal, Clerance$(\boldsymbol{u})$ that favors velocities far from the obstacles, and Velocity$(\boldsymbol{u})$ that favors high speeds. The solution is the control $\boldsymbol{u}_i$ that maximizes this function.

DWA solves the problem in the control space using information of the vehicle dynamics, thus the method is well adapted to work on vehicles with slow dynamic capabilities or that work at high speeds.

47.9.5 Velocity Obstacles

The method of velocity obstacles (VO) solves the problem in two steps, by computing as intermediate information a subset of the $\mathcal{U}$. The framework is equal to that of DWA. The difference is that the computation of the set of safe trajectories $\mathcal{U}_A$ takes into account the velocity of the obstacles, which is described next.

Let $\boldsymbol{v}_i$ be the velocity of obstacle i (which became enlarged with the vehicle radius occupying an area B_i) and $\boldsymbol{u}$ a given vehicle control. The set of colliding relative velocities is called *collision cone*

$$CC_i = \left\{\boldsymbol{u}_i | \lambda_i \bigcap B_i \neq \emptyset\right\}\,, \tag{47.17}$$

where λ_i is the direction of the unitary vector $\boldsymbol{u}_i = \boldsymbol{u}_i - \boldsymbol{v}_i$. The *velocity obstacle* is this set in a common absolute system of reference

$$VO_i = CC_i \oplus \boldsymbol{v}_i\,, \tag{47.18}$$

where $\oplus$ is the Minkowski vector sum. The set of unsafe trajectories is the union of the velocity obstacles for

Avidbots Ex1016
Page 1219 of 2154

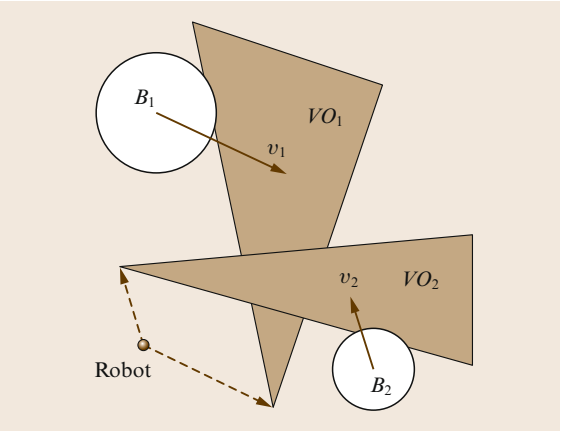


Fig. 47.13 Subset of not safe controls $\bar{\mathcal{U}}_A = VO_1 \cup VO_2$. A control vector out of this velocity generates a collision-free motion with the moving obstacles

each moving obstacle $\bar{\mathcal{U}}_A = \cup_i VO_i$ (Fig. 47.13). The advantage of this method is that takes into account the velocity of the obstacles; thus it is well suited in dynamic scenarios.

47.9.6 Nearness Diagram Navigation

This method is more a methodology to design obstacle avoidance methods rather than a method in itself. Nearness diagram navigation (ND) is an obstacle avoidance method obtained with a geometric implementation following this methodology. The idea behind is to employ a divide and conquer strategy based on situations to simplify the obstacle avoidance problem following the situated-activity paradigm [47.54]. First, there is a set of situations that represent all the cases among robot locations, obstacles, and target locations. Also, for each case there is a motion law associated with it. During the execution phase, at time t_i one situation is identified and the corresponding law is used to compute the motion.

Situations

The situations are represented in a binary decision tree. The selection of a situation depends on the obstacles $\mathcal{O}(\boldsymbol{q}_{t_i})$, on the robot location $\boldsymbol{q}_{t_i}$ and target $\boldsymbol{q}_{\text{target}}$. The criteria are based on high-level entities, like a security distance around the robot bounds and a motion area (that identifies suitable areas of motion). For example, one criterion is if there are obstacles within the security zone. Another is if the motion area is large or narrow. The result is only one situation, since by definition and representation (binary decision tree) the set of situations is complete and exclusive (Fig. 47.14).

Actions

Associated to any situation there is an action that computes the motion to adapt the behavior to the case

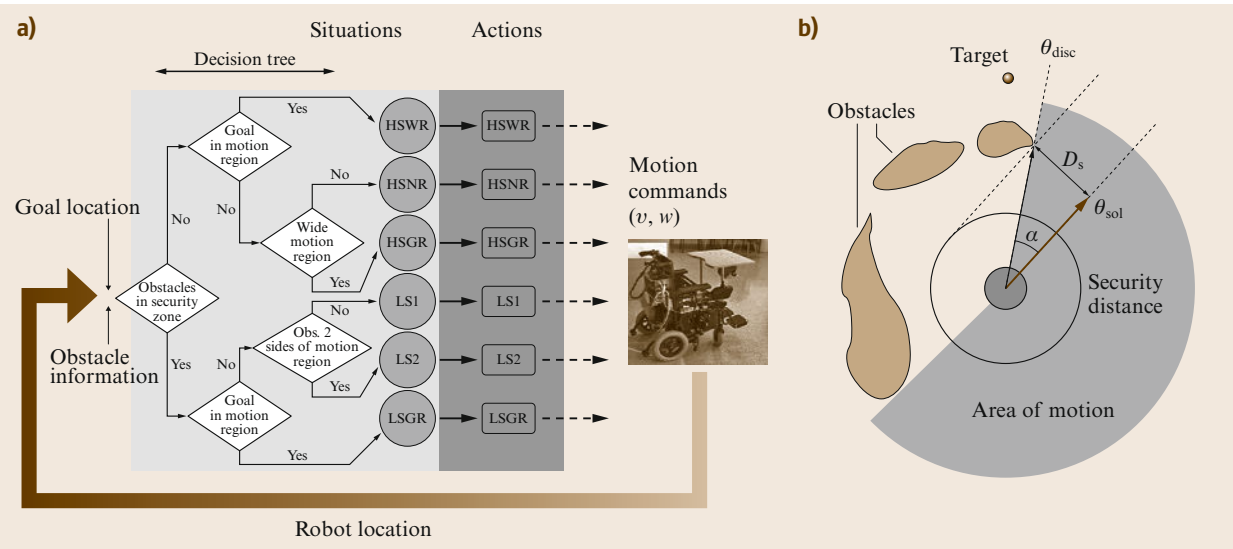


Fig. 47.14 **(a)** Design diagram of the method. Given the obstacle information and the target location, one situation is identified given some criteria. Next the associated action is executed computing the motion. **(b)** Example of computation of the solution of the ND (geometric implementation). The first step is to identify the situation. There are no obstacles closer than the security distance D_s. Next $\boldsymbol{q}_{\text{target}}$ is not within the motion area. Third, the motion area is wide. With these three criteria we identify the current situation, HSWR. In this situation, the associated action computes the control as $\boldsymbol{u}_i = (v_{sol}, \theta_{sol})$, where v_{sol} is the maximum velocity and θ_{sol} is computed as a deviation α from the limit direction θ_{disc} of the motion area closer to the target direction

represented by each situation. At a high level the actions describe the behavior desired in each situation. For example, one situation is when there are no obstacles within a security area and the goal is within the motion area (HSGR). The solution is to *move towards the goal*. Another situation is when there are no obstacles within a security area, and the goal is not within the motion area but is wide (HSWR). The solution is to *move towards the limit of the motion area but clearing the security area of obstacles*.

The interesting aspect of this method is that it employs a *divide and conquer* strategy to solve the navigation. As a consequence, it simplifies the difficulty of the problem. The first advantage is that the methodology is described at a symbolic level and, thus, there are many ways to implement the method. Second, a geometric implementation of this methodology (ND method) has been demonstrated to address difficult navigation cases, which translate in to achieve safe navigation in dense, complex, and difficult scenarios.

47.10 Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance

There are three aspects of the vehicle that have to be taken into account during the obstacle avoidance process: shape, kinematics, and dynamics. The shape with the kinematics is a geometric problem that involves the representation of the vehicle configurations in collision, given the admissible trajectories $\mathcal{Q}_{t,\infty}$. The dynamics involves the accelerations and temporal considerations, and is derived in two aspects: (i) choosing a control reachable in a short period of time T given the current velocity $\dot{\boldsymbol{q}}_{t_i}$ and the maximum accelerations. (ii) Taking the braking distance into account, so that after a control execution, the vehicle can always stop before collision by applying the maximum deceleration (improving safety).

The problem of shape, kinematics, and dynamics in obstacle avoidance has been taken into account from three different points of view:

1. Designing a way to incorporate the constraints within the method (Sect. 47.9.4)
2. Developing techniques that abstract the vehicle aspects from the application of the method [47.55–58]
3. By techniques that break down the problem into subproblems and incorporate the aspects in sequence [47.59–61] after method usage.

47.10.1 Techniques that Abstract Vehicle Aspects

These techniques are based on constructing an abstraction layer between the aspects of the vehicle and the obstacle avoidance method, in such a way that when the method is applied its solutions already take into account these aspects [47.55–58]. We consider here vehicles whose elementary paths obtained under execution of constant controls can be approximated by circular arcs (for example, a differential-drive robot, syncro-drive, or a tri-cycle). To simplify, a control is translational and rotational velocity $\boldsymbol{u} = (v, w)$. The set of reachable controls $\mathcal{U}_{\mathrm{A}}$ given the current velocity $\dot{\boldsymbol{q}}_{t_i} = (v_0, w_0)$ and the maximum accelerations (a_v, a_w) is obtained by (47.14).

Abstraction Construction

For these vehicles, the configuration space CS is three-dimensional. The idea is to construct, centered in the robot at each time t_i, the manifold of the configuration space $ARM(\boldsymbol{q}_{t_i}) \equiv ARM$ defined by elementary circular paths. The function that defines the manifold is

$$\theta = f(x,y) = \begin{cases} \arctan 2\left(x, \dfrac{x^2-y^2}{2y}\right) & \text{if } y \geq 0\,, \\ -\arctan 2\left(x, -\dfrac{x^2-y^2}{2y}\right) & \text{otherwise}\,. \end{cases} \tag{47.19}$$

It is easy to see that function f is differentiable in $\mathbb{R}^2\backslash(0,0)$. Thus $(x, y, f(x,y))$ defines a two-dimensional manifold in $\mathbb{R}^2 \times S^1$ when $(x,y) \in \mathbb{R}^2\backslash(0,0)$. This manifold ARM contains all the configurations that can be reached at each step of the obstacle avoidance.

Next, in the ARM one computes the exact region of the configurations in collision CO_{ARM} given any shape of the robot (i. e., obstacle representation in the manifold). Given an obstacle point (x_p, y_p) and a point of the robot bounds (x_r, y_r), a point (x_s, y_s) of the CO_{ARM} boundary is obtained by

$$\begin{aligned} x_s &= (y_f + x_i)\cdot a\,, \\ y_s &= (y_f - y_i)\cdot a\,, \end{aligned} \tag{47.20}$$

with

$$\begin{aligned} a = \Big\{ \left[\left(y_f^2 - y_i^2\right) + \left(x_f^2 - x_i^2\right)\right]\left[(y_f - y_i)^2 + (x_f - x_i)^2\right] \Big\} \\ \times \Big[(y_f - y_i)^4 + 2\left(x_f^2 + x_i^2\right)(y_f - y_i)^2 \\ + \left(x_f^2 - x_i^2\right)^2 \Big]^{-1}\,. \end{aligned}$$

This result is used to map the robot bounds for all obstacles in the manifold, computing the exact shape of the CO_{ARM}. Next, one compute the nonadmissible configurations CNA_{ARM} in the manifold ARM, which correspond to configurations that once reached at a given

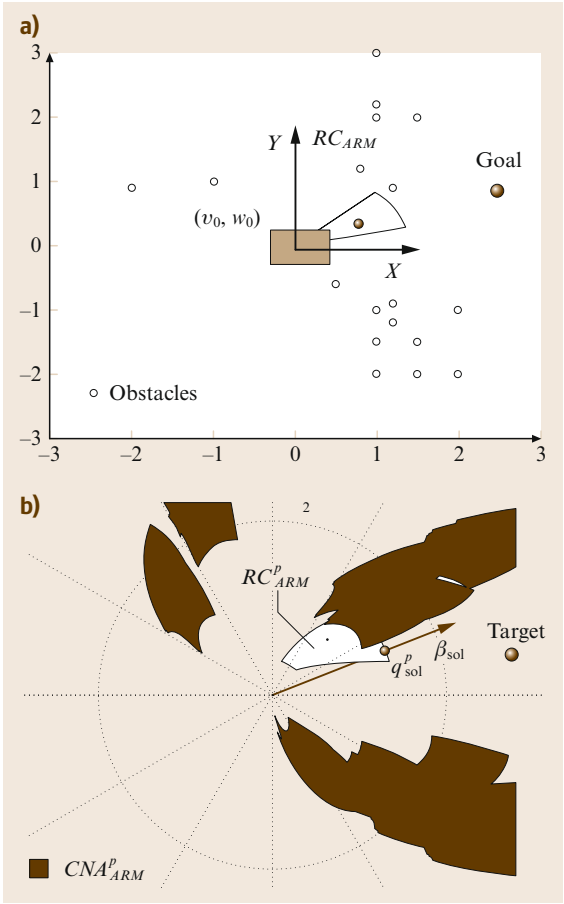


Fig. 47.15 **(a)** Robot location, obstacle information and goal location. **(b)** The abstraction layer in ARM^p. In this representation the obstacles are the nonadmissible region CNA^p_{ARM} in ARM and that the motion is omnidirectional (applicability conditions of many obstacle avoidance methods). The method is applied to obtain the most promising direction β_{sol}, used to obtain the configuration solution $\boldsymbol{q}^p_{sol}$ in the set of reachable configurations RC^p_{ARM}. Finally, the solution is the control $\boldsymbol{u}_{sol}$ that reaches this configuration at time T. This control complies with the kinematics and dynamics and takes into account the exact vehicle shape

velocity in time T, the vehicle cannot be stopped by applying the maximum deceleration without collision (i. e., there is not enough braking distance). These regions CNA_{ARM} are the CO_{ARM} *enlarged* by a magnitude that depends on the maximum vehicle accelerations. The set of configurations RC_{ARM} reachable by controls in a short period of time $\mathcal{U}_A$ is also computed in the ARM. Finally, a change of coordinates is applied to ARM leading to ARM^p. Its effect is that the elementary circular paths become straight segments in the manifold. As a consequence, the problem now is to move an omnidirectional point in a bidimensional space free of any constraint.

Method Application

The final step is to apply the avoidance method on the ARM^p to avoid the CNA^p_{ARM} regions. The method solution β_{sol} is the most promising direction in ARM^p. This direction is used to select and admissible location $\boldsymbol{q}^p_{sol} \notin CNA^p_{ARM}$ in the set of dynamic reachable configurations RC^p_{ARM}. Finally, the control $\boldsymbol{u}_{sol}$ that leads to this location at time T is selected. By construction, this control is kinematically and dynamically admissible, avoids collisions with the exact vehicle shape, and takes into account the braking distance (Fig. 47.15).

Note how with these techniques, the three-tridimensional obstacle avoidance problem is converted into the simple problem of moving a point in a two-dimensional space without kinematic and dynamic restrictions (applicability conditions of many existing methods). Thus, many existing methods can be applied in this abstraction, and as a consequence, the solutions take the vehicle constraints into account.

47.10.2 Techniques of Decomposition in Subproblems

These techniques address the obstacle avoidance problem taking into account the vehicle constraints by breaking down this problem into subproblems:

1. Obstacle avoidance
2. Kinematics and dynamics
3. The shape.

Each subproblem is dealt with in sequence (Fig. 47.16).

Obstacle Avoidance

First the obstacle avoidance method is used by assuming a circular and omnidirectional vehicle. The solution is the most promising motion direction and the velocity $\boldsymbol{u}_1 = (v_1, \theta_1)$ to direct the vehicle towards the target.

Kinematics and Dynamics

Second, this control is converted into a control that complies with the kinematics and dynamics, which tends to align the vehicle with the instantaneous motion direction θ_1 of $\boldsymbol{u}_1$. For example, [47.60, 61] modify the output of the obstacle avoidance method by a feedback action that aligns the vehicle with the direction solution in a least squares fashion. Moreover, [47.59] use a dynamic controller of the vehicle. This controller models the behavior of the vehicle as it would be pulled by a virtual force, and computes the mo-

Steps	Step 1	Step 2	Step 3
Robot aspects	Circular omnidirectional	Circular with kinematics and dynamics	Rectangular with kinematics and dynamics
Subproblem (solution)	Obstacle avoidance	→ Motion controller	→ Shape corrector

Fig. 47.16 Decomposition of the obstacle avoidance problem into subproblems that are addressed in steps, successively incorporating the vehicle aspects

tion that would result after applying this force during a short period of time. The motion generated is the new control. In order to use the controller, the previous control $\boldsymbol{u}_1$ is converted into an instantaneous force $\boldsymbol{F} = (\theta_1, F_{\max}(v_1/v_{\max}))$ input of the controller, which computes a control $\boldsymbol{u}_2$ that complies with the kinematic and dynamics.

Shape

The final step is to assure that the control $\boldsymbol{u}_2$ obtained in the previous step avoids collisions with the exact shape of the vehicle. To do this, a shape corrector is used to check collisions by dynamic simulation of the control. If there is collision, there is a search in the set of reachable controls (47.14) until one of them is collision free. The result of this process is a motion control $\boldsymbol{u}_i$ that guarantees obstacle avoidance and complies with the kinematics and dynamics of the vehicle.

47.11 Integration Planning – Reaction

In this section we show how the obstacle avoidance methods are integrated in real systems. On the one hand, the obstacle avoidance methods are local techniques to address the motion problem. Thus, they are doomed to fall in local minima that translate in trap situations or cyclic motions. This reveals the necessity of a more global reasoning. On the other hand, motion planning techniques compute a geometric path free of collisions, which guarantees global convergence. However, when the scenarios are unknown and evolve, these techniques fail, since the precomputed paths will almost surely collide with obstacles. It seems clear that one key aspect to build a motion system is to combine the best of both worlds: the global knowledge given by motion planning and the reactivity of the obstacle avoidance methods.

The most extended ways to specify the interaction between the deliberation and reaction are: (i) to precompute a path to the target deformed in execution as a function of the changes in the scenario obtained from the sensor information (systems of path deformation), for example, [47.62–67]. (ii) To use a planner at a high frequency with a tactical role, leaving the degree of execution to the reactor [47.68–73].

47.11.1 Systems of Path Deformation

Elastic bands is a method that initially assumes the existence of a geometric path to the target location (computed by a planner). The path is assimilated with a band, subjected to two types of forces: an inner contraction force and an external force. The inner force simulates the tension of a strip and maintains the stress. The external force is exerted by the obstacles and moves the band far from them. During the execution, the new obstacles produce forces that remove the band far from them, guaranteeing their avoidance. These methods are described in Chap. 37.

An extension of the method of path deformation was proposed in [47.64] for nonholonomic systems (VIDEO 80). Even though the objective is the same as for mobile robots without kinematic constraints, avoiding obstacle while following a trajectory, the concepts are completely different. The trajectory Γ of a nonholonomic system is completely defined by an initial configuration $\Gamma(0)$ and the value of the input function $\boldsymbol{u} \in C^1(I, \mathbb{R}^m)$, where I is an interval. The trajectory deformation method for nonholonomic systems is thus based on the perturbation of the input function of the current trajectory in order to achieve three objectives:

1. Keeping the nonholonomic constraints satisfied
2. Getting away from obstacles detected online by onboard sensors
3. Keeping unchanged the initial and last configurations of the trajectory after deformation.

Perturbing the input function of Γ, by a vector-valued input perturbation $\boldsymbol{v} \in C^1(I, \mathbb{R}^m)$, yields a trajectory deformation $\eta \in C^1(I, \mathbb{R}^n)$

$$\boldsymbol{u} \leftarrow \boldsymbol{u} + \tau \boldsymbol{v} \Rightarrow \Gamma \leftarrow \Gamma + \tau \eta \,,$$

where τ is a positive, asymptotically small real number. As an approximation of order 1, the relation between $\boldsymbol{u}$ and η is given by the linearized system, which we do not express here.

Potential Field of Obstacles

Obstacles are taken into account by defining a potential field over the configuration space, which increases when

the robot gets closer to obstacles. This potential field is lifted into the space of trajectories by integration along the trajectory of the configuration potential value.

Discretization of the Input Space

The space $C^1(I, \mathbb{R}^m)$ of input perturbations is an infinite-dimensional vector space. The choice of an input perturbation is restricted to a finite-dimensional subspace spanned by p arbitrary test functions, $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p$ where p is a positive integer. An input perturbation: $\boldsymbol{u} = \sum_{i=1}^{p} \lambda_i \boldsymbol{e}_i$ is thus defined by a vector $\lambda \in \mathbb{R}^p$. The variation of the trajectory potential is linear with respect to λ.

Boundary Conditions

The boundary conditions consisting in applying a trajectory perturbation equal to zero at both ends of interval I is linear with respect to λ. It is, therefore, easy to find a vector λ making the potential decrease and satisfying the boundary conditions.

Nonholonomic Constraint Deviation

The approximation of order 1 in the relation between the input perturbation and trajectory deformation induces a side effect: after a few iterations, the nonholonomic constraints are no longer satisfied. This side effect is corrected by considering an augmented system with n control vector fields $f_1, \ldots, f_n$ spanning $\mathbb{R}^n$ for each configuration and by keeping the input components u_{m+1} to u_n along the additional vector fields as close as possible to 0. Figure 47.17 shows an example of the trajectory deformation algorithm applied to a nonholonomic system.

47.11.2 Systems of Tactical Planning

The systems of tactical planning recompute at a high frequency a path to the target location, and use the main course to advise the obstacle avoidance module. The design of these motion systems involves at least the synthesis of three functionalities. The construction of a model, the deliberative planning, and the obstacle avoidance. The modeler constructs a representation base for deliberation and memory for the reactive behavior. The planner generates global plans used tactically to guide the obstacle avoidance module, which generates the local motion. Next we give a perspective of the three functionalities and three possible tools to implement them [47.73].

Model Builder Module

Construction of a model of the environment (to increase the spatial domain of the planning and used as local memory for obstacle avoidance). One possibility is to use a binary occupancy grid that is updated whenever a new sensory measurement is available, and to employ a scan matching technique [47.74, 75] to improve the vehicle odometry before integrating any new measure in the grid.

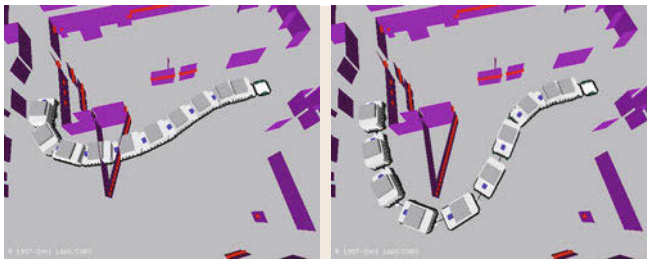


Fig. 47.17 A differentially-driven mobile robot towing a trailer applying the trajectory deformation algorithm to avoid obstacles detected online

Planner Module

Extraction of the connectivity of the free space (used to avoid the cyclical motions and trap situations). A good choice is a dynamic navigation function such as the D* [47.76, 77]. The idea behind the planner is to focus the search locally in the areas where the changes in the scenario structure have occurred and affect the computation of the path. The planner avoids the local minima and is computationally very efficient for real-time implementations.

Obstacle Avoidance Module

Computation of the collision-free motion. Any of the methods described in this chapter could be used. One possibility is the ND method (Sect. 47.9.6), since it has been demonstrated to be very effi-

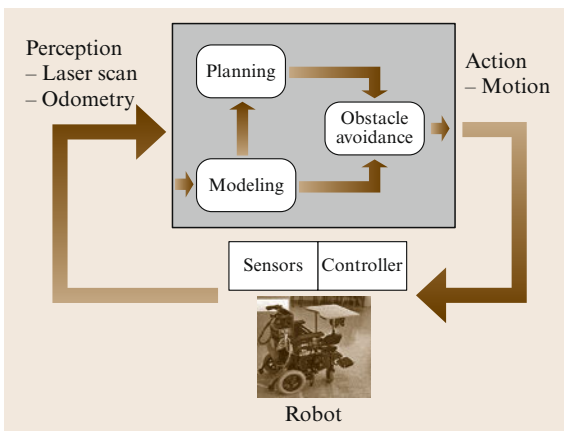


Fig. 47.18 Overview of the system of motion

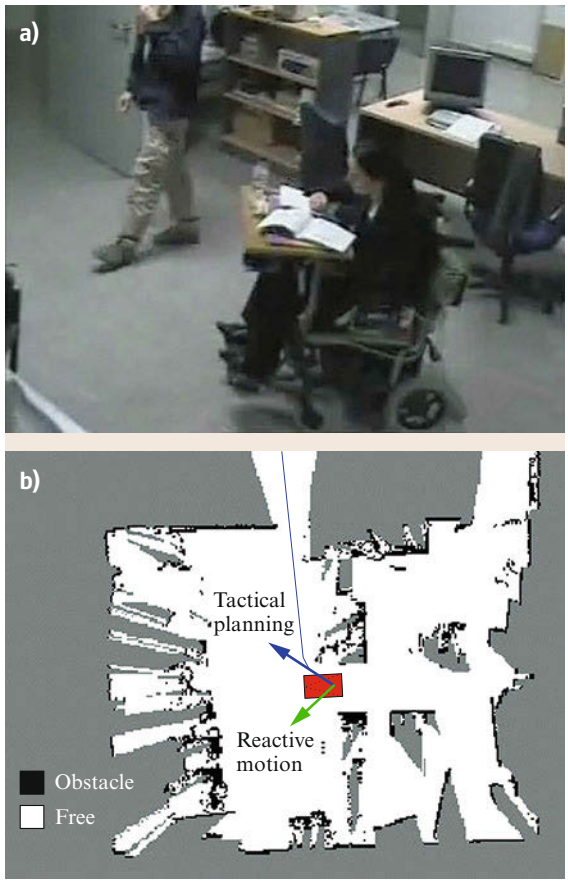


Fig. 47.19 **(a)** Snapshot of an experiment carried out with the motion system described working on a wheelchair robot equipped with a planar range laser sensor. **(b)** Information of the motion system at a given time: current accumulated map of the scenario, path computed with the planner and tactical direction of motion, and solution of the reactive obstacle avoidance method ◄

cient and robust in environments with little space to manoeuvre.

Globally the system works as follows (Fig. 47.18): given a laser scan and the odometry of the vehicle, the model builder incorporates this information into the existing model. Next, the information of the changes in obstacle and free space in the model is used by the planner module to compute the course to follow to reach the goal. Finally, the avoidance module uses the information of the obstacles contained in the grid and information of this tactical planner to generate the motion (to drive the vehicle free of collisions towards the goal). The motion is executed by the vehicle controller and the process restarts with a new sensorial measurement. It is important to stress that the three modules should work synchronously within the perception – action cycle for consistency reasons. The advantage of these systems is that the synergy of the modules allows to avoid trap situations and the cyclic motions (limitations associated to the local nature of the avoidance methods). Figure 47.19 shows a snapshot of an experiment carried out with this motion system.

47.12 Conclusions, Future Directions, and Further Reading

The algorithmic tools presented in this chapter show that motion planning and obstacle avoidance research techniques have reached a level of maturity that allow their transfer onto real platforms (VIDEO 710, VIDEO 714). Today, several indoor mobile robots use obstacle avoidance techniques on a daily basis to guide visitors in museums. Outdoor applications still require some developments in perception and modeling. For these applications, 3-D sensing capabilities are necessary to model the environment but also to detect obstacles. As an example, several car manufacturer are working on parallel parking assistance. The difficult point of this application is to build a model of a parking spot from 3-D data in a great variety of environments.

The main challenge of autonomous motion in mobile robots currently consists in integrating techniques from different robotics research domains in order to plan and execute motions for very complex systems, like humanoid robots, for instance. Integration in the sense that different software components need to work together on a machine but also in a scientific meaning: the classical motion planning formulation with configuration variables that locate the robot with respect to a global reference frame does not fit partially known environments with imprecise maps. Robotic tasks like motion need to be specified with respect to landmarks of the environment. For instance, grasping an object is by definition a motion specified with respect to the position of the object. Very little has been done to design a general framework in this direction.

For useful complementary reading on motion planning and obstacle avoidance for mobile robots see [47.5, 78–80].

Avidbots Ex1016
Page 1225 of 2154

Video-References

VIDEO 80 Sensor-based trajectory deformation and docking for nonholonomic mobile robots
available from http://handbookofrobotics.org/view-chapter/47/videodetails/80

VIDEO 707 Autonomous robotic smart wheelchair navigation in an urban environment
available from http://handbookofrobotics.org/view-chapter/47/videodetails/707

VIDEO 708 Sena wheelchair: Autonomous navigation at University of Malaga (2007)
available from http://handbookofrobotics.org/view-chapter/47/videodetails/708

VIDEO 709 Robotic wheelchair: Autonomous navigation with Google Glass
available from http://handbookofrobotics.org/view-chapter/47/videodetails/709

VIDEO 710 A ride in the google self driving car
available from http://handbookofrobotics.org/view-chapter/47/videodetails/710

VIDEO 711 Mobile robot navigation system in outdoor pedestrian environment
available from http://handbookofrobotics.org/view-chapter/47/videodetails/711

VIDEO 712 Mobile robot autonomous navigation in Gracia district, Barcelona
available from http://handbookofrobotics.org/view-chapter/47/videodetails/712

VIDEO 713 Autonomous navigation of a mobile vehicle
available from http://handbookofrobotics.org/view-chapter/47/videodetails/713

VIDEO 714 Autonomous robot cars drive DARPA Urban challenge
available from http://handbookofrobotics.org/view-chapter/47/videodetails/714

References

47.1 N.J. Nilson: A mobile automaton: An application of artificial intelligence techniques, 1st Int. Jt. Conf. Artif. Intell. (1969) pp. 509–520

47.2 A. Thompson: The navigation system of the JPL robot, 5th Int. Jt. Conf. Artif. Intell., Cambridge (1977) pp. 749–757

47.3 G. Giralt, R. Sobek, R. Chatila: A multi-level planning and navigation system for a mobile robot: A 1st approach to Hilare, 6th Int. Jt. Conf. Artif. Intell. (1979) pp. 335–337

47.4 T. Lozano-Pérez: Spatial planning: A configuration space approach, IEEE Trans. Comput. **32**(2), 108–120 (1983)

47.5 J.C. Latombe: *Robot Motion Planning* (Kluwer, Dordrecht 1991)

47.6 J.-P. Laumond: Feasible trajectories for mobile robots with kinematic and environment constraints. In: *Intelligent Autonomous Systems*, ed. by F.C.A. Groen (Hertzberger, Amsterdam 1987) pp. 346–354

47.7 Z. Li, J.F. Canny: *Nonholonomic Motion Planning* (Kluwer, Dordrecht 1992)

47.8 M. Likhachev, D. Ferguson: Planning long dynamically-feasible maneuvers for autonomous vehicles, Int. J. Robotics Res. **28**(8), 933–945 (2009)

47.9 H. Sussmann: Lie brackets, real analyticity and geometric control. In: *Differential Geometric Control Theory*, Progress in Mathematics, Vol. 27, ed. by R. Brockett, R. Millman, H. Sussmann (Birkhauser, New York 1982) pp. 1–116

47.10 H.J. Sussmann, V. Jurdjevic: Controllability of nonlinear systems, J. Differ. Equ. **12**, 95–116 (1972)

47.11 J.-P. Laumond: Singularities and topological aspects in nonholonomic motion planning. In: *Nonholonomic motion Planning*, ed. by Z. Li, J.F. Canny (Kluwer, Boston 1992) pp. 149–199

47.12 J.-P. Laumond, J.J. Risler: Nonholonomic systems: Controllability and complexity, Theor. Comput. Sci. **157**, 101–114 (1996)

47.13 J.-P. Laumond, P. Jacobs, M. Taix, R. Murray: A motion planner for nonholonomic mobile robot, IEEE Trans. Robotics Autom. **10**(5), 577–593 (1994)

47.14 P. Cheng, S.M. LaValle: Reducing metric sensitivity in randomized trajectory design, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 43–48

47.15 J.A. Reeds, R.A. Shepp: Optimal paths for a car that goes both forward and backwards, Pac. J. Math. **145**(2), 367–393 (1990)

47.16 P. Souères, J.-P. Laumond: Shortest path synthesis for a car-like robot, IEEE Trans. Autom. Control **41**(5), 672–688 (1996)

47.17 D. Balkcom, M. Mason: Time optimal trajectories for bounded velocity differential drive vehicles, Int. J. Robotics Res. **21**(3), 199–218 (2002)

47.18 D. Tilbury, R. Murray, S. Sastry: Trajectory generation for the *n*-trailer problem using Goursat normal form, IEEE Trans. Autom. Control **40**(5), 802–819 (1995)

47.19 S. Sekhavat, J.-P. Laumond: Topological property for collision-free nonholonomic motion planning: The case of sinusoidal inputs for chained form systems, IEEE Trans. Robotics Autom. **14**(5), 671–680 (1998)

47.20 M. Fliess, J. Lévine, P. Martin, P. Rouchon: Flatness and defect of non-linear systems: Introductory theory and examples, Int. J. Control **61**(6), 1327–1361 (1995)

47.21 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness and motion planning: The car with *n* trailers, Eur. Control Conf. (1993) pp. 1518–1522

47.22 F. Lamiraux, J.-P. Laumond: Flatness and small-time controllability of multibody mobile robots:

Avidbots Ex1016
Page 1226 of 2154

Application to motion planning, IEEE Trans. Autom. Control **45**(10), 1878–1881 (2000)
47.23 P. Rouchon: Necessary condition and genericity of dynamic feedback linearization, J. Math. Syst. Estim. Control **4**(2), 1–14 (1994)
47.24 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness, motion planning and trailer systems, IEEE Int. Conf. Decis. Control (1993) pp. 2700–2705
47.25 S. Sekhavat, J. Hermosillo: Cycab bi-steerable cars: A new family of differentially flat systems, Adv. Robotics **16**(5), 445–462 (2002)
47.26 J. Barraquand, J.C. Latombe: Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles, Algorithmica **10**, 121–155 (1993)
47.27 A. Divelbiss, T. Wen: A path space approach to nonholonomic motion planning in the presence of obstacles, IEEE Trans. Robotics Autom. **13**(3), 443–451 (1997)
47.28 S. LaValle, J. Kuffner: Randomized kinodynamic planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 473–479
47.29 F. Lamiraux, E. Ferré, E. Vallée: Kinodynamic motion planning: Connecting exploration trees using trajectory optimization methods, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 3987–3992
47.30 V. Lumelsky, A. Stepanov: Path planning strategies for a point mobile automation moving admist unknown obstacles of arbitrary shape, Algorithmica **2**, 403–430 (1987)
47.31 R. Chatila: Path planning and environmental learning in a mobile robot system, Eur. Conf. Artif. Intell. (1982)
47.32 L. Gouzenes: Strategies for solving collision-free trajectories problems for mobile robots and manipulator robots, Int. J. Robotics Res. **3**(4), 51–65 (1984)
47.33 R. Chattergy: Some heuristics for the navigation of a robot, Int. J. Robotics Res. **4**(1), 59–66 (1985)
47.34 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robotics Res. **5**, 90–98 (1986)
47.35 B.H. Krogh, C.E. Thorpe: Integrated path planning and dynamic steering control for autonomous vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 1664–1669
47.36 J. Borenstein, Y. Koren: Real-time obstacle avoidance for fast mobile robots, IEEE Trans. Syst. Man Cybern. **19**(5), 1179–1187 (1989)
47.37 K. Azarm, G. Schmidt: Integrated mobile robot motion planning and execution in changing indoor environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 298–305
47.38 A. Masoud, S. Masoud, M. Bayoumi: Robot navigation using a pressure generated mechanical stress field, the biharmonical potential approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 124–129
47.39 L. Singh, H. Stephanou, J. Wen: Real-time robot motion control with circulatory fields, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 2737–2742
47.40 J. Borenstein, Y. Koren: The vector field histogram-fast obstacle avoidance for mbile robots, IEEE Trans. Robotics Autom. **7**, 278–288 (1991)
47.41 J. Minguez: The obstacle restriction method (ORM): Obstacle avoidance in difficult scenarios, IEEE/RSJ Int. Conf. Intell. Robot Syst. (IROS) (2005)
47.42 D. Vikerimark, J. Minguez: Reactive obstacle avoidance for mobile robots that operate in confined 3-D workspaces, IEEE Mediterr. Electrotech. Conf. (2006)
47.43 W. Feiten, R. Bauer, G. Lawitzky: Robust obstacle avoidance in unknown and cramped environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 2412–2417
47.44 D. Fox, W. Burgard, S. Thrun: The dynamic window approach to collision avoidance, IEEE Robotics Autom. Mag. **4**(1), 23–33 (1997)
47.45 P. Fiorini, Z. Shiller: Motion planning in dynamic environments using velocity obstacles, Int. J. Robotics Res. **17**(7), 760–772 (1998)
47.46 F. Large, C. Laugier, Z. Shiller: Navigation among moving obstacles using the NLVO: Principles and applications to intelligent vehicles, Auton. Robots **19**(2), 159–171 (2005)
47.47 R. Simmons: The curvature-velocity method for local obstacle avoidance, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 3375–3382
47.48 J. Minguez, L. Montano: Nearness diagram (ND) navigation: Collision avoidance in troublesome scenarios, IEEE Trans. Robotics Autom. **20**(1), 45–59 (2004)
47.49 J. Minguez, J. Osuna, L. Montano: A divide and conquer strategy to achieve reactive collision avoidance in troublesome scenarios, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)
47.50 J.W. Durham, F. Bullo: Smooth nearness-diagram navigation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008)
47.51 C.-C. Yu, W.-C. Chen, C.-C. Wang: Self-tuning nearness diagram navigation, Int. Conf. Serv. Interact. Robotics (SIRCon) (2009)
47.52 M. Mujahad, D. Fischer, B. Mertsching, H. Jaddu: Closest gap based (CG) reactive obstacle avoidance Navigation for highly cluttered environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 1805–1812
47.53 R.B. Tilove: Local obstacle avoidance for mobile robots based on the method of artificial potentials, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 566–571
47.54 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1999)
47.55 J. Minguez, L. Montano: Abstracting any vehicle shape and the kinematics and dynamic constraints from reactive collision avoidance methods, Eur. Conf. Mob. Robots (2007)
47.56 J. Minguez, L. Montano, J. Santos-Victor: Abstracting the vehicle shape and kinematic constraints from the obstacle avoidance methods, Auton. Robots **20**(1), 43–59 (2006)
47.57 L. Armesto, V. Girbés, M. Vincze, S. Olufs, P. Muñoz-Benavent: Mobile robot obstacle avoidance based

Avidbots Ex1016
Page 1227 of 2154

on quasi-holonomic smooth paths, Lect. Notes Comput. Sci. **7429**, 244–255 (2012)
47.58 J.L. Blanco, J. González, J.A. Fernández-Madrigal: Foundations of parameterized trajectories-based space transformations for obstacle avoidance. In: *Motion Planning*, ed. by X.-Y. Jing (InTech, Rijeka 2008)
47.59 J. Minguez, L. Montano: Robot navigation in very complex dense and cluttered indoor/outdoor environments, 15th IFAC World Congr. (2002)
47.60 A. De Luca, G. Oriolo: Local incremental planning for nonholonomic mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 104–110
47.61 A. Bemporad, A. De Luca, G. Oriolo: Local incremental planning for car-like robot navigating among obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 1205–1211
47.62 S. Quinlan, O. Khatib: Elastic bands: Connecting path planning and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 802–807
47.63 O. Brock, O. Khatib: Real-time replanning in high-dimensional configuration spaces using sets of homotopic paths, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 550–555
47.64 F. Lamiraux, D. Bonnafous, O. Lefebvre: Reactive path deformation for nonholonomic mobile robots, IEEE Trans. Robotics **20**(6), 967–977 (2004)
47.65 Y. Yang, O. Brock: Elastic roadmaps – Motion generation for autonomous mobile manipulation, Auton. Robots **28**(1), 113–130 (2010)
47.66 E. Yoshida, C. Esteves, I. Belousov, J.-P. Laumond, T. Sakaguchi, K. Yokoi: Planning 3-d collision-free dynamic robotic motion through iterative reshaping, IEEE Trans. Robotics **24**, 1186–1198 (2008)
47.67 H. Kurniawati, T. Fraichard: From path to trajectory deformation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)
47.68 O. Brock, O. Khatib: High-speed navigation using the global dynamic window approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 341–346
47.69 I. Ulrich, J. Borenstein: VFH*: Local obstacle avoidance with look-ahead verification, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 2505–2511
47.70 J. Minguez, L. Montano: Sensor-based motion robot motion generation in unknown, dynamic and troublesome scenarios, Robotics Auton. Syst. **52**(4), 290–311 (2005)
47.71 C. Stachniss, W. Burgard: An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 508–513
47.72 R. Philipsen, R. Siegwart: Smooth and efficient obstacle avoidance for a tour guide robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)
47.73 L. Montesano, J. Minguez, L. Montano: Lessons learned in integration for sensor-based robot navigation systems, Int. J. Adv. Robotics Syst. **3**(1), 85–91 (2006)
47.74 F. Lu, E. Milios: Robot pose estimation in unknown environments by matching 2-D range scans, Intell. Robotic Syst. **18**, 249–275 (1997)
47.75 J. Minguez, L. Montesano, F. Lamiraux: Metric-based iterative closest point scan matching for sensor displacement estimation, IEEE Trans. Robotics **22**(5), 1047–1054 (2006)
47.76 A. Stenz: The focussed *d** Algorithm for real-time replanning, Int. Jt. Conf. Artif. Intell. (IJCAI) (1995) pp. 1652–1659
47.77 S. Koenig, M. Likhachev: Improved fast replanning for robot navigation in unknown terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)
47.78 J.-P. Laumond: Robot motion planning and control. In: *Lecture Notes in Control and Information Science*, ed. by J.P. Laumond (Springer, New York 1998)
47.79 H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun: *Principles of Robot Motion* (MIT Press, Cambridge 2005)
47.80 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2006)

Avidbots Ex1016
Page 1228 of 2154

Multimedia Contents

48. Modeling and Control of Legged Robots

Pierre-Brice Wieber, Russ Tedrake, Scott Kuindersma

The promise of legged robots over wheeled robots is to provide improved mobility over rough terrain. Unfortunately, this promise comes at the cost of a significant increase in complexity. We now have a good understanding of how to make legged robots walk and run dynamically, but further research is still necessary to make them walk and run efficiently in terms of energy, speed, reactivity, versatility, and robustness. In this chapter, we will discuss how legged robots are usually modeled, how their stability analysis is approached, how dynamic motions are generated and controlled, and finally summarize the current trends in trying to improve their performance. The main problem is avoiding to fall. This can prove difficult since legged robots have to rely entirely on available contact forces to do so. The temporality of leg motions appears to be a key aspect in this respect, as current control solutions include continuous anticipation of future motion (using some form of model predictive control), or focusing more specifically on limit cycles and orbital stability.

48.1 **A Brief History of Legged Robots** 1204
48.2 **The Dynamics of Legged Locomotion** 1204
48.2.1 Lagrangian Dynamics 1205
48.2.2 Newton and Euler Equations of Motion 1205
48.2.3 Contact Models 1207
48.3 **Stability Analysis – Not Falling Down** 1209
48.3.1 Fixed Points 1210
48.3.2 Limit Cycles 1210
48.3.3 Viability 1211
48.3.4 Controllability 1212
48.3.5 Robust or Stochastic Stability 1213
48.3.6 Input-Output Stability 1214
48.3.7 Stability Margins 1214
48.4 **Generation of Dynamic Walking and Running Motions** 1214
48.4.1 Early Offline Motion Generation Schemes 1215
48.4.2 Online Motion Generation: A Model Predictive Control Point of View 1215
48.4.3 Motion in Constrained Environments 1219
48.4.4 Motion Generation with Limited Computing Resources 1220
48.5 **Motion and Force Control** 1222
48.6 **Towards More Efficient Walking** 1225
48.6.1 Gait Generation for Dynamic Walking 1225
48.6.2 Orbital Trajectory Stabilization and Control 1226
48.7 **Different Contact Behaviors** 1227
48.7.1 Wall Climbing 1227
48.7.2 Tethered Walking 1227
48.7.3 Legs with Wheels 1228
48.7.4 Wheels with Legs 1228
48.8 **Conclusion** 1228
References 1228

Part E | 48

The promise of legged robots over standard wheeled robots is to provide improved mobility over rough terrain. This promise builds on the decoupling between the environment and the main body of the robot that the presence of articulated legs allows, with two consequences. First, the motion of the main body of the robot can be made largely independent from the roughness of the terrain, within the kinematic limits of the legs: legs provide an active suspension system. Indeed, one of the most advanced hexapod robots of the 1980s was aptly called the Adaptive Suspension Vehicle [48.1]. Second, this decoupling allows legs to temporarily leave their

contact with the ground: isolated footholds on a discontinuous terrain can be overcome, allowing to visit places absolutely out of reach otherwise. Note that having feet firmly planted on the ground is not mandatory here: skating is an equally interesting option, although rarely approached so far in robotics.

Unfortunately, this promise comes at the cost of a hindering increase in complexity. It is only with the unveiling of the Honda P2 humanoid robot in 1996 [48.2], and later of the Boston Dynamics BigDog quadruped robot in 2005 that legged robots finally began to deliver real-life capabilities that are just beginning to match the long sought animal-like mobility over rough terrain. Not matching yet the capabilities of humans and animals, legged robots do contribute however already to understanding their locomotion, as evidenced by the many fruitful collaborations between robotics and biomechanics researchers.

48.1 A Brief History of Legged Robots

Before the advent of digital computers, legged machines could be approached only by electromechanical means, lacking in advanced feedback control. This *prerobotics* period culminated with the General Electric Walking Truck developed by Ralph Mosher, which inspired awe in the mid 1960s. The limb motions of this elephant-size quadruped machine were directly reflecting the limb motions of the onboard operator, who was responsible for all motion control and synchronization. Unfortunately, the strenuous concentration that this required limited operation to less than 15 minutes.

Digitally controlled legged robots started to appear in the late 1960s. Among early pioneers, Robert McGhee initiated a series of quadruped and hexapod robots first at University of South California, then at Ohio State University, culminating in the mid 1980s with the Adaptive Suspension Vehicle, a human carrying hexapod vehicle walking on natural and irregular outdoor terrain [48.1], while Ichiro Kato initiated a long series of biped and humanoid robots in the Waseda University, a series still continuing nearly half a century later [48.3]. But by the end of the 1970s, all legged robots were still limited to quasi-static gaits, i. e., slow walking motions with the center of mass of the robot always kept above its feet.

The transition to dynamic legged locomotion occurred in the beginning of the 1980s, with the first dynamically walking bipedal robot demonstrated at the Tokyo University [48.4], and the famous series of hopping and running monopedal, bipedal and quadrupedal robots developed at the MIT (Massachusetts Institute of Technology) LegLab under the direction of *Marc Raibert* [48.5]. Key theoretical breakthroughs came in the end of the 1980s, when *Tad McGeer* demonstrated that stable dynamic walking motions could be obtained by pure mechanical means, giving rise to a whole new field of research, *passive dynamic walking*, introducing new, key concepts such as orbital stability using Poincaré maps [48.6], with one simple conclusion: you need not have complete (or any) control to be able to walk dynamically and efficiently.

Legged robots were still mostly research laboratory curiosities working in limited situations when Honda unveiled the P2 humanoid robot in 1996 [48.2], a decade long secret project demonstrating unprecedented versatility and robustness, followed in 2000 by the Asimo humanoid robot. The world of humanoid and legged robots was ripe for companies to begin investing. A handful of other Japanese companies such as Toyota or Kawada were quick to follow with their own humanoid robots, while Sony began selling more than 150 000 of its Aibo home companion robot dogs. Boston Dynamics, a company *Marc Raibert* founded after leaving the MIT LegLab, finally unveiled its BigDog quadruped robot in 2005 [48.7], which was the first to demonstrate true animal-like locomotion capabilities on rough terrain.

The progress over the last decades has been remarkable. Profound questions have finally been answered: we now understand how to make legged robots walk and run dynamically. But other profound questions still have to be answered, such as how best to make them walk and run *efficiently*. The performance of legged robots needs to be improved in many ways: energy, speed, reactivity, versatility, robustness, etc. We will therefore discuss in this chapter how legged robots are usually modeled in Sect. 48.2 and how dynamic motions are currently generated and controlled in Sections 48.4 and 48.5, before discussing in Sect. 48.6 the current trends in improving their efficiency.

48.2 The Dynamics of Legged Locomotion

One of the major difficulties in making a legged robot walk or run is keeping its balance: where should the robot place its feet, how should it move its body in order to move safely in a given direction, even in case

of strong perturbations? This difficulty comes from the fact that contact forces with the environment are an absolute necessity to generate and control locomotion, but they are limited by the mechanical laws of unilateral contact.

This essential role of the contact forces is particularly clear in the derivatives of the total linear and angular momenta of the robot, the former involving the motion of its center of mass. Because of the importance of contact forces for legged locomotion, we briefly discuss here their different models.

48.2.1 Lagrangian Dynamics

Structure of the Configuration Space

As for every robot moving in their 3-D environment (in space or underwater for example), the configuration space of legged robots combines the configuration $\hat{\boldsymbol{q}} \in \mathbb{R}^N$ of their N joints with a global position $\boldsymbol{x}_0 \in \mathbb{R}^3$ and orientation $\boldsymbol{\theta}_0 \in \mathbb{R}^3$ (representing an element of $SO(3)$)

$$\boldsymbol{q} = \begin{pmatrix} \hat{\boldsymbol{q}} \\ \boldsymbol{x}_0 \\ \boldsymbol{\theta}_0 \end{pmatrix} . \tag{48.1}$$

The position $\boldsymbol{x}_0$ and orientation $\boldsymbol{\theta}_0$ are typically those of a central body (pelvis or trunk) or of an extremity (foot or hand).

Structure of the Lagrangian Dynamics

The specific structure of the configuration space outlined above is naturally reflected in the Lagrangian dynamics

$$\mathbf{M}(\boldsymbol{q}) \left[\begin{pmatrix} \ddot{\hat{\boldsymbol{q}}} \\ \ddot{\boldsymbol{x}}_0 \\ \ddot{\boldsymbol{\theta}}_0 \end{pmatrix} + \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{g} \\ \boldsymbol{0} \end{pmatrix} \right] + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix} + \sum_i \boldsymbol{C}_i(\boldsymbol{q})^{\mathrm{T}} \boldsymbol{f}_i \tag{48.2}$$

of the system, where $\mathbf{M}(\boldsymbol{q}) \in \mathbb{R}^{(N+6)\times(N+6)}$ is the generalized inertia matrix of the robot, $-\boldsymbol{g} \in \mathbb{R}^3$ is the constant gravity acceleration vector, $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{N+6}$ is the vector of Coriolis and centrifugal effects, $\boldsymbol{u} \in \mathbb{R}^N$ is the vector of joint torques, and for all i, $\boldsymbol{f}_i \in \mathbb{R}^3$ is a force exerted by the environment on the robot and $\mathbf{C}_i(\boldsymbol{q}) \in \mathbb{R}^{(N+6)\times 3}$ is the associated Jacobian matrix [48.8].

Since the vector $\boldsymbol{u}$ of joint torques has the same size as the vector $\hat{\boldsymbol{q}}$ of joint positions, the whole dynamics including the global position $\boldsymbol{x}_0$ and orientation $\boldsymbol{\theta}_0$ appears to be underactuated if no external forces $\boldsymbol{f}_i$ are exerted.

48.2.2 Newton and Euler Equations of Motion

Center of Mass and Angular Momentum

A consequence of the structure (48.2) of the Lagrangian dynamics is that the part of this dynamics which is not directly actuated involves the Newton and Euler equations of motion of the robot taken as a whole ([48.8] for detailed derivations). The Newton equation can be written in the following way

$$m\,(\ddot{\boldsymbol{c}} + \boldsymbol{g}) = \sum_i \boldsymbol{f}_i \,, \tag{48.3}$$

with m the total mass of the robot and $\boldsymbol{c}$ the position of its center of mass (COM). The Euler equation can be expressed with respect to the COM in the following way

$$\dot{\mathbf{L}} = \sum_i (\boldsymbol{p}_i - \boldsymbol{c}) \times \boldsymbol{f}_i \,, \tag{48.4}$$

with $\boldsymbol{p}_i$ the points of applications of the forces $\boldsymbol{f}_i$ and

$$\mathbf{L} = \sum_k (\boldsymbol{x}_k - \boldsymbol{c}) \times m_k \dot{\boldsymbol{x}}_k + \mathbf{I}_k \boldsymbol{\omega}_k \tag{48.5}$$

the angular momentum of the whole robot with respect to its COM, with $\dot{\boldsymbol{x}}_k$ and $\boldsymbol{\omega}_k$ the translation and rotation velocities of the different parts k of the robot, m_k and $\mathbf{I}_k$ their masses and inertia tensor matrices (expressed in global coordinates).

The Newton equation makes it obvious that the robot needs external forces $\boldsymbol{f}_i$ in order to move its COM in a direction other than that of gravity. The Euler equation is more subtle, as we will see during flight phases.

Flight Phases

During flight phases, when a legged robot is not in contact with its environment, not experiencing any contact forces $\boldsymbol{f}_i$, the Newton equation (48.3) simplifies to

$$\ddot{\boldsymbol{c}} = -\boldsymbol{g} \,. \tag{48.6}$$

In this case, the COM invariably accelerates along the gravity vector $-\boldsymbol{g}$ with constant horizontal speed, following a standard falling motion: there is absolutely no possibility to control the COM to move in any different way. The Euler equation (48.4) simplifies in the same way to

$$\dot{\mathbf{L}} = 0 \,, \tag{48.7}$$

imposing a conservation of the angular momentum $\mathbf{L}$. In this case, however, the robot is still able to generate and control both joint motions and global rotations, this is how cats are able to fall back on their feet when

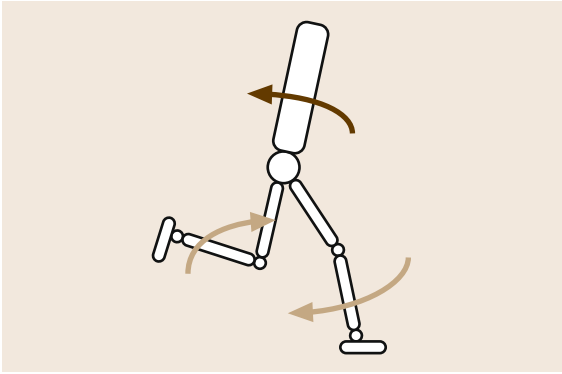


Fig. 48.1 Even though the angular momentum is constant during flight phases, the robot is still able to generate and control rotations of the whole body with the help of leg or arm motions (*light brown*), as a result of the *nonholonomy* of the angular momentum. This is how cats fall back on their feet when dropped from any initial orientation

dropped from any initial orientation (Fig. 48.1). This is a result of the *nonholonomy* of the angular momentum (48.5) which is not the derivative of any function of the configuration of the robot [48.9]. As a result, even though the angular momentum $\mathbf{L}$ is kept constant during the whole flight phase, the joint configuration $\hat{\boldsymbol{q}}$, and the global orientation $\boldsymbol{\theta}_0$ of the robot can be driven to any desired value at the end of the flight phase. We will see in Sect. 48.5 how this impacts the control of legged robots. Note that the dynamics of legged robots during flight phases is similar to the dynamics of free-floating space robots discussed in Chap. 55. Further discussion and developments can be found there.

In Contact with a Flat Ground: The Center of Pressure

In case the forces applied by the environment on the robot are due to contacts with a flat ground (while standing still, walking or running), let us consider a reference frame oriented along the ground, with the z axis orthogonal to it (therefore tilted if the ground is tilted, see Fig. 48.5). Without loss of generality, let us suppose that the points of contact, $\boldsymbol{p}_i$, with the ground are all such that $p_i^z = 0$.

Let us consider then the sum of the Euler (48.4) and the cross product of the COM $\boldsymbol{c}$ with the Newton (48.3)

$$m\boldsymbol{c} \times (\ddot{\boldsymbol{c}} + \boldsymbol{g}) + \dot{\mathbf{L}} = \sum_i \boldsymbol{p}_i \times \boldsymbol{f}_i \,, \tag{48.8}$$

and let us divide the result by the z coordinate of the Newton equation to obtain

$$\frac{m\boldsymbol{c} \times (\ddot{\boldsymbol{c}} + \boldsymbol{g}) + \dot{\mathbf{L}}}{m(\ddot{c}^z + g^z)} = \frac{\sum_i \boldsymbol{p}_i \times \boldsymbol{f}_i}{\sum_i f_i^z} \,. \tag{48.9}$$

Since $p_o^z = 0$, the x and y coordinates of this equation can be simplified in the following way

$$\boldsymbol{c}^{x,y} - \frac{c^z}{\ddot{c}^z + g^z}(\ddot{\boldsymbol{c}}^{x,y} + \boldsymbol{g}^{x,y}) + \frac{1}{m(\ddot{c}^z + g^z)}\mathbf{S}\dot{\mathbf{L}}^{x,y} = \frac{\sum_i f_i^z \boldsymbol{p}_i^{x,y}}{\sum_i f_i^z} \,, \tag{48.10}$$

with a simple rotation matrix

$$\mathbf{S} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} .$$

On the right hand side of (48.10) appears the definition of the center of pressure (COP) $\boldsymbol{z}$ of the contact forces, $\boldsymbol{f}_i$. These contact forces are usually unilateral (the robot can push on the ground, not pull)

$$f_i^z \geq 0 \,, \tag{48.11}$$

which implies that the CoP is bound to lie in the convex hull of the contact points (Fig. 48.2)

$$\boldsymbol{z}^{x,y} = \frac{\sum_i f_i^z \boldsymbol{p}_i^{x,y}}{\sum_i f_i^z} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} \,. \tag{48.12}$$

Combining this inclusion with the dynamic (48.10) reveals an ordinary differential inclusion (ODI)

$$\boldsymbol{c}^{x,y} - \frac{c^z}{\ddot{c}^z + g^z}(\ddot{\boldsymbol{c}}^{x,y} + \boldsymbol{g}^{x,y}) + \frac{1}{m(\ddot{c}^z + g^z)}\mathbf{S}\dot{\mathbf{L}}^{x,y} = \boldsymbol{z}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} \,, \tag{48.13}$$

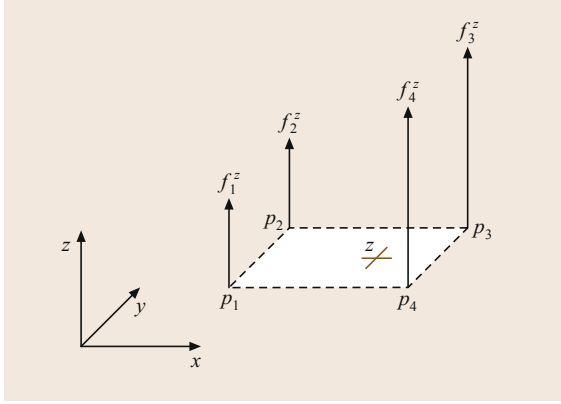


Fig. 48.2 The CoP $\boldsymbol{z}$ is bound to lie in the convex hull of contact points $\boldsymbol{p}_i$

Avidbots Ex1016
Page 1232 of 2154

which bounds the motion of the COM $\boldsymbol{c}$, of the robot and the variations of its angular momentum $\mathbf{L}$ with respect to the position $\boldsymbol{p}_i^{x,y}$ of the contact points.

This ODI can be reorganized in the following way

$$\frac{c^z}{\ddot{c}^z+g^z}(\ddot{\boldsymbol{c}}^{x,y}+\boldsymbol{g}^{x,y})=(\boldsymbol{c}^{x,y}-\boldsymbol{z}^{x,y})+\frac{1}{m(\ddot{c}^z+g^z)}\mathbf{S}\dot{\mathbf{L}}^{x,y}\,, \tag{48.14}$$

in order to expose its simple geometric meaning in Fig. 48.3. We can see especially that aside from the effects of gravity $\boldsymbol{g}^{x,y}$ and variations $\dot{\boldsymbol{L}}^{x,y}$ of the angular momentum, the horizontal acceleration $\ddot{\boldsymbol{c}}^{x,y}$ of the COM is the result of a force pushing the COM $\boldsymbol{c}^{x,y}$ away from the CoP $\boldsymbol{z}^{x,y}$, which is bound to lie in the convex hull of the contact points. We have here an intrinsically unstable dynamics.

Note finally that the definition (48.12) of the CoP can be reorganized to show that the horizontal momenta of the contact forces $\boldsymbol{f}_i$ with respect to the CoP $\boldsymbol{z}$ are equal to zero

$$\left[\sum_i(\boldsymbol{p}_i-\boldsymbol{z})\times\boldsymbol{f}_i\right]^{x,y}=\sum_i(\boldsymbol{p}_i^{x,y}-\boldsymbol{z}^{x,y})f_i^z=0\,. \tag{48.15}$$

Hence the CoP is also referred to as the zero moment point (ZMP) [48.10, 11].

In Contact with Multiple Surfaces

If the contact points $\boldsymbol{p}_i$ are not all on the same plane, we can introduce a CoP for each contact surface, but we cannot introduce a unique CoP for all contact forces as we did previously. Approximations, and generalizations to limited multiple contact situations have been proposed but haven't been widely adopted [48.11–14]. In the general case, the Newton and Euler equations (48.3) and (48.4) have to be considered explicitly together with the unilaterality condition (48.11) in order to check which motion is feasible or not [48.15–17]. The problem we have to solve then relates very closely to the *force closure* problem discussed in Chap. 38 on grasping. Different ways to solve it as quickly as possible have been proposed [48.15, 18], and used mostly so far to measure offline the stability robustness of a given motion [48.19, 20], and only recently for motion planning, once again offline [48.21]. Refinements to curved contact surfaces have also been proposed [48.22].

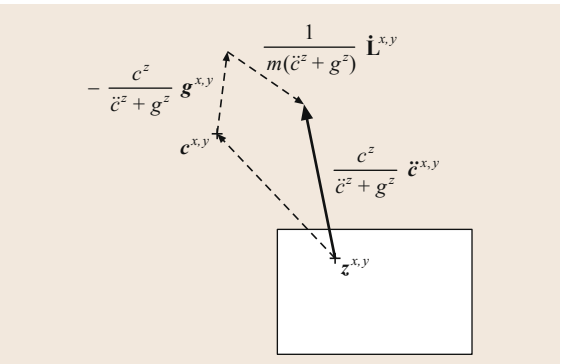


Fig. 48.3 The horizontal acceleration $\ddot{\boldsymbol{c}}^{x,y}$ of the COM of the robot is the sum of a force pushing the COM $\boldsymbol{c}^{x,y}$ away from the CoP $\boldsymbol{z}^{x,y}$ the effect of gravity $-\boldsymbol{g}^{x,y}$ and variations $\dot{\mathbf{L}}^{x,y}$ of the angular momentum

48.2.3 Contact Models

The structure of the Lagrangian dynamics makes it clear that contact forces are central to the modeling and control of legged robots. But note that the only characteristics of these forces that we have introduced so far is their unilaterality (48.11). As a result, the previous analysis applies to various contact situations: walking and running motions, but also sliding situations such as when skiing or skating, or even rolling situations [48.23, 24].

Let us briefly discuss now standard contact models (more details can be found in Chap. 37 on contact modeling and manipulation). Concerning motion and forces tangential to the contact surfaces, a simple Coulomb friction model is usually considered. Concerning motion and forces orthogonal to the contact surfaces, two options are generally considered: a compliant or a rigid model, introducing impacts and other nonsmooth behaviors.

Coulomb Friction

When a contact point $\boldsymbol{p}_i$ is sliding on its contact surface, the corresponding tangential contact force $\boldsymbol{f}_i^{x,y}$ is proportional to the normal force f_i^z in a direction opposite to the sliding motion

$$\boldsymbol{f}_i^{x,y}=-\mu_0 f_i^z\frac{\dot{\boldsymbol{p}}_i^{x,y}}{\|\dot{\boldsymbol{p}}_i^{x,y}\|}\ \text{if}\ \dot{\boldsymbol{p}}_i^{x,y}\neq 0\,, \tag{48.16}$$

with $\mu_0>0$ the friction coefficient. When the contact point is sticking and not sliding, the norm of the tangential force is simply bounded, with the same friction coefficient

$$\|\boldsymbol{f}_i^{x,y}\|\leq\mu_0 f_i^z\ \text{if}\ \dot{\boldsymbol{p}}_i^{x,y}=0\,. \tag{48.17}$$

This is typically referred to as the *friction cone*. Note that this friction model directly implies the unilaterality condition (48.11).

Compliant Contact Models

Compliant contact models take into account the viscoelastic properties of the materials in contact in the direction orthogonal to the contact surfaces

$$f_i^z = -K_i p_i^z - \Lambda_i \dot{p}_i^z \text{ if } p_i^z \leq 0\,, \tag{48.18}$$

where K_i and Λ_i are stiffness and damping coefficients, respectively. In this case, the normal force f_i^z appears to be the result of a penetration of the contact point below the contact surface, i. e., when $p_i^z < 0$. Note that, as it is here, this model doesn't satisfy the unilaterality condition (48.11) in all situations and needs therefore to be saturated to enforce this property. Of course, when there is no contact, there is no contact force

$$f_i^z = 0 \text{ if } p_i^z > 0\,. \tag{48.19}$$

A similar spring-damper model can also be used to determine the frictional contact force $f_i^{x,y}$, subject to the bounds of the friction cone (48.17).

Rigid Contact Models

Rigid contact models are somewhat simpler to introduce: either there is contact and the normal force can take any nonnegative value,

$$f_i^z \geq 0 \text{ if } p_i^z = 0\,, \tag{48.20}$$

or there is no contact and no contact force

$$f_i^z = 0 \text{ if } p_i^z > 0\,. \tag{48.21}$$

In this case, no penetration of the contact points below the contact surfaces are considered. Note also that the unilaterality condition (48.11) is satisfied here by definition. As a matter of fact, this rigid model can be summarized in the following way

$$f_i^z \geq 0,\ p_i^z \geq 0,\ f_i^z p_i^z = 0\,, \tag{48.22}$$

where it appears that both the normal force f_i^z and the position of the contact point with respect to the contact surface p_i^z must be nonnegative, but at least one of them must be equal to 0. This is called a *complementarity condition*. Analogous to (48.20) and (48.21), it defines the normal force to be either zero, or the force necessary to maintain $p_i^z = 0$. Note that the sticking-or-sliding behavior of Coulomb friction can also be represented with such a complementarity condition. Note also that this set of implicit equations may not have a solution or may have multiple ones in certain pathologic situations [48.25].

Rigid and compliant contact models are compared in Fig. 48.4 in a static situation (when $\dot{p}_i^z = 0$), showing

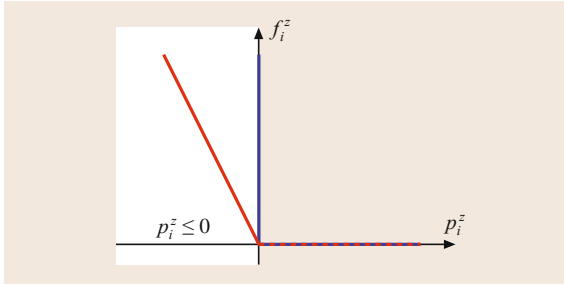


Fig. 48.4 Comparison of a rigid (*blue*) and compliant (*red*) contact model in a static situation. Unlike the rigid model, the compliant model considers a penetration of the contact point below the contact surface, when $p_i^z < 0$. The graph of the rigid model, with the specific shape of a right angle, is called a Signorini graph

clearly how the rigid model corresponds to an infinitely stiff compliant model. Compliant contact models can model more accurately the deformation of the contacting bodies, but accurate models often require very stiff springs, causing the resulting differential equations to be numerically stiff, what can slow down or complicate numerical analysis. Rigid models are comparatively more straightforward to integrate and analyze, with purely rigid bodies and contact surfaces. They are therefore preferred when theoretically or numerically studying the stability of legged robots, when optimizing walking or running trajectories, etc.

Impacts

Since rigid contact models do not allow penetration of the contact points below the contact surfaces, when a point $\boldsymbol{p}_i$ reaches a contact surface $p_i^z = 0$ with a velocity $\dot{p}_i^z < 0$, this velocity has to change instantaneously in order to satisfy the no-penetration assumption, $p_i^z \geq 0$ [48.26]. What happens in this situation is an *impact*, a discontinuity of the velocity, where we need to distinguish the velocity of the robot before impact $\dot{\boldsymbol{q}}^-$, and its velocity after impact, $\dot{\boldsymbol{q}}^+$. A straightforward integration of the Lagrangian dynamics (48.2) over a time singleton [48.27] gives us a relation between pre- and post-impact velocities and impulsive forces $\boldsymbol{F}_i$

$$\mathbf{M}(\boldsymbol{q})\left(\dot{\boldsymbol{q}}^+ - \dot{\boldsymbol{q}}^-\right) = \sum_i \mathbf{C}_i(\boldsymbol{q})^{\mathrm{T}} \boldsymbol{F}_i\,. \tag{48.23}$$

But in order to completely define the impact law and compute the post-impact velocity, we also need a model of the impulsive forces. Unfortunately, this is a complex problem still open to research [48.28, 29], especially in situations of multiple contacts which are the standard situation for legged robots. The usual approach for

legged robots is therefore to assume the post-impact behavior, usually considering that the contact points eventually stick to the contact surfaces

$$\dot{\boldsymbol{p}}_i^+ = \mathbf{C}_i \dot{\boldsymbol{q}}^+ = 0\,. \tag{48.24}$$

This is, however, a strong assumption, and reality can be much more complex, as demonstrated in [48.30]. Moreover, a recent study shows that post-impact sticking may not always be desirable [48.31].

Impacts are central in most stability analyses of passive dynamic walking machines (Sect. 48.6), but they can be potentially destructive for the mechanical parts of a robot. As a result, they are often carefully avoided and therefore generally disregarded outside of passive dynamic walkers. A recent study shows however that this is probably a mistake [48.32].

Hybrid and Nonsmooth Dynamics

With rigid contacts, the dynamics of legged robots appears to switch, depending on the contact situations (48.20) or (48.21). Discontinuities of the state of the robot also occur at impacts. A classical way to combine these different aspects is with a hybrid dynamical system. This approach has however some limitations [48.33, 34], the most obvious one being its incapacity to handle properly *Zeno behaviors*, infinite accumulations of impacts in finite time [48.35]. An example of such a situation is when a legged robot is rigidly standing on one foot, and gently rocking from one contact edge to the opposite contact edge, with a continuously decreasing period (at least according to the perfectly rigid model).

Impacts with multiple contacts and Zeno behaviors are not the only difficulties with rigid contact models: there is also the Painlevé paradox, tangential impacts, impacts without collisions, etc. The nonsmooth dynamics approach [48.25, 27, 33, 34], where the Lagrangian dynamics is turned into a measure differential equation, accelerations being abstract measures, velocities being functions with locally bounded variations, appears to be a much more adequate way to deal with all these intricacies. But it has not been widely adopted for legged robots because of a significant increase in mathematical complexity.

Many of the complexities of rigid body contact dynamics can be avoided by discretizing the system dynamics in time and reasoning about the integral of contact forces acting over a time step. In particular, no distinction is made between impulses and finite contact forces over a time step. By conservatively approximating the friction cone as a polyhedron, forward dynamics can be cast as a linear complementarity problem (LCP) [48.36], for which efficient solvers exist. This has become a popular formulation for simulation of rigid bodies in frictional contact and it has recently seen applications to control design for legged systems [48.37].

48.3 Stability Analysis – Not Falling Down

The robot model described above represents a complex, constrained, nonlinear dynamical system. Our goal for this section is to understand the long-term behavior of this dynamical system, to answer questions like, *Will the robot fall down?* As we will see in the following sections, this long-term behavior can be changed by open-loop or closed-loop feedback control, with the explicit goal of minimizing the likelihood that the robot will fall down.

For controlled nonlinear dynamical systems, there are a number of useful concepts relating to their safeness and stability, including:

- *Fixed points*: Stable fixed points represent the static postures in which the robot can safely stand still.
- *Limit cycles*: Limit cycles provide a natural extension of fixed-point analysis to *periodic* walking or running motions.
- *Viability*: Viability is a concept of controlled invariance, which analyzes the set of states from which the robot is able to avoid to fall. Unfortunately, this property can be intractable to compute.
- *Controllability*: Controllability provides a slightly restricted notion of viability, analyzing the set of states from which the robot is capable of returning to a particular fixed point (or limit cycle). This can be more tractable to compute than viability, especially for simple models.

In addition, if there are unknown errors in the robot model, if there is uncertainty about the environment (e.g., the location of the ground), or if there are unmodeled disturbances, then additional tools are available from robustness analysis and stochastic stability:

- *Robust stability*: Robust stability (or viability) examines the properties of the system considering worst-case (bounded) disturbances. For instance, a robust controller may be able to guarantee that a fixed point is stable even if the estimate of the mass of the trunk is wrong by $\pm 10\%$.

- *Stochastic stability*: Stochastic analysis provides tools to investigate the *probability* of falling down. For many robot disturbance models, the system will always fall eventually (with probability one), but analysis can reveal long-living *metastable* distributions.
- *Input-output stability*: This analysis treats a particular disturbance as an input, a performance criteria as output, and attempts to compute a relative *gain* or *sensitivity* of the robot performance due to this input.
- *Stability margins*: Robustness analysis can be difficult. In practice control designers often settle for the system staying comfortably away from the boundaries of deterministic stability.

We give a very brief overview of these tools in the remainder of this section, with an attempt to emphasize features of the analysis that are particular to legged robots.

48.3.1 Fixed Points

Given a first-order ordinary differential equation governed by $\dot{\boldsymbol{x}} = \boldsymbol{a}(\boldsymbol{x})$, a *fixed point* of the system is a state $\boldsymbol{x}^*$, for which $\boldsymbol{a}(\boldsymbol{x}^*) = 0$. The governing equations for a legged robot, given in Sect. 48.2, are second-order, take a control input, and potentially require a differential inclusion to describe the contact forces, but the notion of a fixed point is still meaningful. Here a fixed point is a configuration $\boldsymbol{q}^*$, for which there exists feasible control inputs $\boldsymbol{u}^*$, and feasible contact forces $\boldsymbol{f}^*$, so that $\dot{\boldsymbol{q}}^* = \ddot{\boldsymbol{q}}^* = \boldsymbol{0}$ is a solution to (48.2). A fixed point of this system is a posture in which the legged robot is able to stand still.

In static situations, when $\ddot{\boldsymbol{c}} = \dot{\mathbf{L}} = \boldsymbol{0}$, the ODI (48.13) gives the following necessary condition

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\boldsymbol{g}^{x,y} = \boldsymbol{z}^{x,y} \in \text{conv}\left\{p_i^{x,y}\right\} \tag{48.25}$$

for a static equilibrium on a flat ground. This necessary condition states that the COM $\boldsymbol{c}$ must project on the ground along the gravity vector $-\boldsymbol{g}$, inside the convex hull of the contact points p_i, also known as the *support polygon* (Fig. 48.5). This is a very well known and widely used necessary condition. Note, however, that it is valid only when considering contacts with a flat ground [48.15].

Once a fixed point has been found, one would often like to examine its stability. For smooth nonlinear differential equations, local stability can often be established via linearizing the dynamics at the fixed point and then applying well-known tools from linear systems theory. Indeed, these tools can also be applied

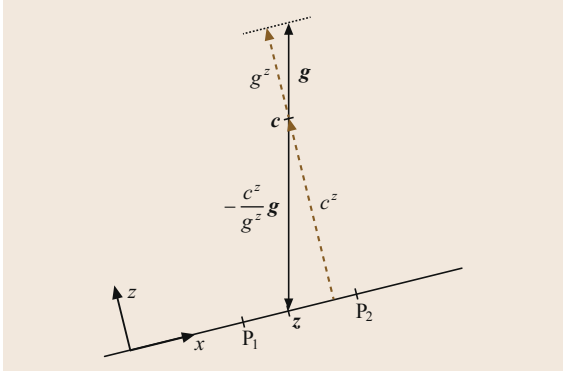


Fig. 48.5 A necessary static equilibrium condition on a flat ground is that the COM $\boldsymbol{c}$ must project on the ground along the gravity vector $-\boldsymbol{g}$ inside the convex hull of the contact points P_i also known as the *support polygon* (*red*)

here, *under the strong assumption that the contact conditions do not change* (e.g., either $f_i^z = 0$ and $p_i^z \geq 0$, or $f_i^z \geq 0$ and $p_i^z = 0$). Each active contact adds an equality constraint to the linearized system and the eigenvalues of this constrained linear system are only meaningful in the minimal coordinates; e.g., for a robot standing on horizontal ground the assumption precludes the ability to examine a perturbation of the entire robot in the vertical direction. Note that in a hybrid model of the legged robot, this assumption is equivalent to linearizing within a single hybrid mode, and it is common practice to describe each hybrid mode in its minimal coordinates [48.38]. Once local stability is established, it may also be possible to understand the *region of attraction* of the fixed point [48.39, 40]. Evaluating the stability of these models through changes in contact configurations is a new and important area of research. For recent work, see for instance [48.41].

48.3.2 Limit Cycles

A natural extension of fixed-point analysis to walking and running motions is to examine the existence and stability of periodic orbits, or limit cycles, of the dynamical system. Here a periodic orbit is a solution $\{\langle \boldsymbol{q}(t), \boldsymbol{u}(t), \boldsymbol{f}(t)\rangle \mid t \in [0, \infty)\}$ of the dynamical system with a finite period $T > 0$ such that $\boldsymbol{q}(t+T) = \boldsymbol{q}(t), \boldsymbol{u}(t+T) = \boldsymbol{u}(t)$, and $\boldsymbol{f}(t+T) = \boldsymbol{f}(t)$. Almost always, periodic orbits are discovered numerically using techniques from motion planning like those described in Sect. 48.4. For passive, or very minimally actuated robots, the very existence of a periodic solution can be exciting [48.42]; robots with more actuators typically have an abundance of periodic solutions and planning techniques can be used to find solutions that, for instance, minimize the a quantity of interest such as the

cost of transport [48.43] or a measure of open-loop stability [48.44].

A periodic solution $\langle \boldsymbol{q}_0(\cdot), \boldsymbol{u}_0(\cdot), \boldsymbol{f}_0(\cdot) \rangle$ with period T, is (asymptotically) *orbitally stable* [48.45] or *limit-cycle stable* if, given initial conditions $\boldsymbol{q}(0)$ we have

$$\lim_{t \to \infty} \left[\min_{0 \le t' < T} \| \boldsymbol{q}(t) - \boldsymbol{q}_0(t') \| \right] = 0 \,. \tag{48.26}$$

The key feature here that there is no requirement that system trajectories converge in time; only the distance to the closest point on the orbit must go to zero. This stability can be defined for initial conditions restricted to a local neighborhood of $\boldsymbol{q}_0(\cdot)$, for a region containing $\boldsymbol{q}_0(\cdot)$, or globally (though no interesting legged robot is globally stable) and for inputs determined by a variety of open- or closed-loop control policies, $\boldsymbol{u}(t) = \boldsymbol{\pi}(t, \boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{f}(t))$. Under mild conditions, a sufficient criteria for establishing orbital stability is establishing fixed-point stability of a Poincaré map [48.46], as illustrated in Fig. 48.6. Due to the limitation that finding periodic solutions is typically done numerically, Poincaré map analysis is also typically done numerically, e.g., using finite differences to evaluate a local linearization of the map. This does not necessarily preclude rigorous analysis, and techniques exist to compute regions of attraction to hybrid limit cycles despite potential numerical inaccuracies [48.47]. Finally we note that limit cycle analysis can be applied (carefully) to solutions that are not periodic in all states – for instance, the x, y location of the floating base should be left out if the robot needs to make forward progress!

The primary advantage of limit cycle analysis is that, using the Poincaré map and related methods, many of the tools from fixed point analysis and linear systems theory can be easily extended to evaluate (local) stability and even to design a (locally) stabilizing controller. However, this is a very limited definition of locomotion; useful locomotion through nontrivial environments will likely require aperiodic motions.

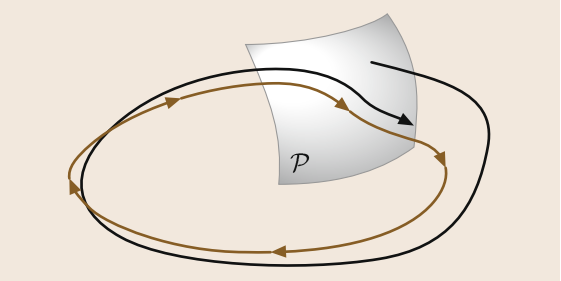


Fig. 48.6 Illustration of a Poincaré surface $\mathcal{P}$ defined on a periodic orbit (*red*) and a trajectory converging to the orbit (*black*)

Extensions to Aperiodic Stability

The notion of stability requires convergence to some nominal solution as time goes to infinity, thus the nominal solution must be defined over an infinite interval. A common approach involves stitching together periodic solutions with provably bounded finite-time transition maneuvers which can switch between them [48.48]. Orbital stability of an (infinite) aperiodic trajectory can be established using a transverse linearization or *moving Poincaré section* [48.49]. For real-world situations where the entire desired trajectory cannot be known apriori, these guarantees can also be provided using online motion planning and receding-horizon control [48.50].

48.3.3 Viability

If the objective of a legged robot is simply to avoid falling down, it can be possible then to directly define (at least theoretically) the set of postures $\mathcal{F} \subseteq \mathbb{R}^{N+6}$, where the robot has fallen down and that the robot should simply avoid. This leads to introducing the set of states from which the robot is able to avoid entering $\mathcal{F}$ – the set of states from which the robot is able to avoid falling down. These states will be called *viable*, following the *viability* theory developed in [48.51] for general ODIs and introduced in the analysis of walking robots in [48.52]. In principle, the set of all viable states represents a safe operating envelope for the robot. It is typically intractable to compute explicitly but it can be approached indirectly in the case of walking robots in the following way.

Center of Mass Dynamics During Horizontal Walking

In case the robot is walking on a horizontal ground, the z axis is aligned with gravity, so $\boldsymbol{g}^{x,y} = 0$ and the ODI (48.13) becomes

$$\boldsymbol{c}^{x,y} - \frac{c^z}{\ddot{c}^z + g^z} \ddot{\boldsymbol{c}}^{x,y} + \frac{1}{m(\ddot{c}^z + g^z)} \mathbf{S}\dot{\mathbf{L}}^{x,y} \in \operatorname{conv}\left\{ \boldsymbol{p}_i^{x,y} \right\} . \tag{48.27}$$

We can observe that it is linear with respect to the horizontal motion $\boldsymbol{c}^{x,y}$, $\ddot{\boldsymbol{c}}^{x,y}$ of the COM and variations of the angular momentum $\mathbf{L}^{x,y}$. The vertical motion c^z, $\ddot{c}^z$ of the COM can usually be bounded, but in order to simplify the following derivations, let us suppose that the COM moves strictly horizontally above the ground, i. e., c^z is constant and $\ddot{c}^z = 0$, so the ODI becomes

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z} \ddot{\boldsymbol{c}}^{x,y} + \frac{\mathbf{S}\dot{\mathbf{L}}^{x,y}}{mg^z} \in \operatorname{conv}\left\{ \boldsymbol{p}_i^{x,y} \right\} . \tag{48.28}$$

Variations of the angular momentum $\mathbf{L}$ can also be bounded in the x and y directions, but to make things

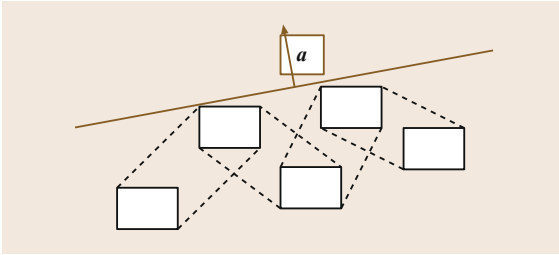


Fig. 48.7 A situation where, for some reason, no steps can be undertaken beyond a certain line (*red*), with $\boldsymbol{a}$ a vector orthogonal to this line and pointing beyond it

as simple as possible, let us consider these variations to be equal to 0, so the ODI takes a very simple second order linear form

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} = \boldsymbol{z}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} . \tag{48.29}$$

An Example of Inevitable Fall

Let us consider furthermore a situation where, for some reason (e.g., a cliff), no contact with the ground can be realized beyond a certain line (Fig. 48.7), with $\boldsymbol{a}$ a vector orthogonal to this line and pointing beyond it. If the COM of the robot reaches this line at a time t_0 with a speed pointing beyond it ($\boldsymbol{a}^{\mathrm{T}}\dot{\boldsymbol{c}}^{x,y}(t_0) > 0$), this linear ODI can be integrated analytically very simply and lead to the following inequality [48.53], valid at all time $t \geq t_0$

$$\boldsymbol{a}^{\mathrm{T}}\left[\boldsymbol{c}^{x,y}(t) - \boldsymbol{c}^{x,y}(t_0)\right] \geq \frac{\boldsymbol{a}^{\mathrm{T}}\dot{\boldsymbol{c}}^{x,y}(t_0)}{\omega}\sinh\left[\omega(t-t_0)\right] , \tag{48.30}$$

with

$$\omega = \sqrt{\frac{g^z}{c^z}} . \tag{48.31}$$

The right hand side is increasing exponentially with time. As a result, the position $\boldsymbol{c}^{x,y}$ of the COM is diverging exponentially in the direction of the vector $\boldsymbol{a}$, leading inexorably to a fall.

A Sufficient Condition for Viability

In the simplified linear case developed above, the inequality (48.30) establishes that in the case of a fall, the motion of the COM diverges exponentially, so the integral of the norm of any n^{th} derivative of its position,

$$\int_{t_0}^{\infty} \left\| \boldsymbol{c}^{(n)}(t) \right\| \mathrm{d}t , \tag{48.32}$$

would be infinite. Therefore, if we can find a finite value for such an integral from a given initial state of the robot, we can conclude that this state is viable.

48.3.4 Controllability

For any dynamical system, an initial condition $\boldsymbol{x}_0$ is *controllable* to a final state $\boldsymbol{x}_f$, if there exists a feasible input trajectory $\boldsymbol{u}(\cdot)$, which takes the system from $\boldsymbol{x}_0$ to $\boldsymbol{x}_f$. An initial state $\boldsymbol{x}_0$ is denoted *finite-time* controllable to $\boldsymbol{x}_f$ if the minimum time required is finite. Controllability analysis is a well understood topic for linear dynamical systems, where a simple rank condition on the *controllability matrix* provides necessary and sufficient conditions to establish whether or not any initial condition can be driven to any final state [48.54]. For nonlinear systems, the property is dependent on both the initial and final states.

In the case of legged robots, instead of considering the set of all viable states introduced earlier, we could consider instead the more limited set of states that are controllable to stable fixed-points – states that can come to a stable stop after a given number of steps. This is a sufficient condition for viability. Such states have been called *capturable* [48.55], and arguably encompass most of the states of interest for legged robots. A variant would be to consider states from which the robot is able to reach a stable limit cycle, or any other state which is known to be viable in the first hand.

The Capture Point

In the simple linear case developed above, the capturable states can be identified analytically with the help of the compound variable

$$\boldsymbol{\xi} = \boldsymbol{c} + \frac{1}{\omega}\dot{\boldsymbol{c}} , \tag{48.33}$$

introduced independently as the *extrapolated center of mass* (XCOM) [48.56], the *capture point* [48.57] or the *divergent component* of the dynamics [48.58]. These three denominations correspond to three key properties of this variable. First of all, a trivial reformulation gives

$$\dot{\boldsymbol{c}} = \omega(\boldsymbol{\xi} - \boldsymbol{c}) , \tag{48.34}$$

revealing that this point $\boldsymbol{\xi}$ is the point where the COM is converging to, hence the extrapolated COM. Following the linear dynamics (48.29), the horizontal motion of this point satisfies

$$\dot{\boldsymbol{\xi}}^{x,y} = \omega(\boldsymbol{\xi}^{x,y} - \boldsymbol{z}^{x,y}) , \tag{48.35}$$

where we can see that this point $\boldsymbol{\xi}$ diverges away from the CoP $\boldsymbol{z}$. But if this point is above the support polygon,

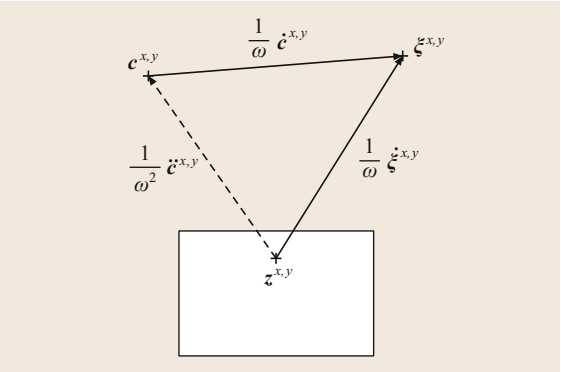


Fig. 48.8 The speed of the COM $\boldsymbol{c}$ points towards the capture point $\boldsymbol{\xi}$, while the speed of this capture point points away from the CoP, $\boldsymbol{z}$. The combination of these two first-order dynamics gives the acceleration $\ddot{\boldsymbol{c}}$, pushing the COM away from the CoP

if we have

$$\boldsymbol{\xi}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} , \tag{48.36}$$

we can have $\boldsymbol{z}^{x,y} = \boldsymbol{\xi}^{x,y}$ so the point $\boldsymbol{\xi}$ can stay motionless, and the COM will converge to it and come to a stop, hence it can be seen as a capture point (CP). Finally, we can observe that the second-order linear dynamics (48.29) has been decomposed here as two first-order linear dynamics (48.34) and (48.35), the first one being stable, with the COM converging to the XCOM, the second one being unstable, with the XCOM diverging away from the CoP, hence the divergent part of the dynamics (48.29). Here appears an interesting structure relating the motion of the points $\boldsymbol{z}$, $\dot{\boldsymbol{c}}$ and $\boldsymbol{\xi}$, shown in Fig. 48.8 [48.59, 60].

An important observation here is that if the state of the robot satisfies the condition (48.36), then the robot can stop without making any step and it is capturable and viable. Further analysis, considering any given number of steps and a nonzero angular momentum L can be found in [48.60].

48.3.5 Robust or Stochastic Stability

For the deterministic robot dynamics models described in Sect. 48.2, the concepts of fixed-points, limit cycles, viability, and controllability form the core issues of analysis. However, in the presence of disturbances, such as model inaccuracies or unpredicted variation in terrain, tools from robust verification must be called upon to characterize the stability of a system. Broadly speaking, robust stability analysis exists in two forms. In *worst-case* analysis, an upper bound is given for a disturbance and the goal is to demonstrate that for any disturbance within this bound, the system will not reach an unstable state. Alternatively, *stochastic* analysis can be performed where a disturbance probability distribution is specified with the goal being to demonstrate that the system avoids unstable states with high probability. Worst-case analysis has received surprisingly little attention to date in the legged robotics community, however analysis approaches based on Lyapunov functions [48.38, 41, 47] can be easily extended using the notion of a *common Lyapunov function* [48.61]. One challenge here is that the nominal solutions, e.g., the exact shape of the nominal limit cycle, are often parameter dependent: the system might still be stable with different parameters, but to a slightly different limit cycle.

Stochastic stability has received relatively more attention in the legged robotics community, perhaps because many interesting models of possible disturbances do result in robots that walk for a long time, but do eventually fall over. Such systems cannot be classified as stable, but it seems incomplete to simply call them unstable. Such systems are said to be *metastable*, a concept first established in the physics community [48.62] and later used to analyze walking systems [48.63]. From this point of view, the stability of a system can be characterized by its mean time to failure.

Recall the idea of Poincaré maps described above, but now consider the discrete time stochastic return map dynamics for a periodic walking system,

$$\boldsymbol{x}_{k+1} = r(\boldsymbol{x}_k, h_k) , \tag{48.37}$$

where $h_k \sim P(h)$ is a random variable that, e.g., captures the terrain height at time step k. Discretizing the state space $\mathcal{X} \subseteq \mathbb{R}^{(N+6)} \times \mathbb{R}^{(N+6)}$ into d states allows us to define the Markov process based on the above return map

$$\boldsymbol{p}_{k+1} = \mathbf{T}\boldsymbol{p}_k , \tag{48.38}$$

where $p_k \in \mathbb{R}^d$ is the state probably distribution vector and $\mathbf{T}$ is a stochastic matrix with entries $T_{ij} = \Pr(\boldsymbol{x}_{k+1}^i | \boldsymbol{x}_k^j)$. If we assume that all failure states are absorbing and can be grouped together into a single state, this corresponds to a column in $\mathbf{T}$ containing a single 1 and $\mathrm{d}-1$ zeros. The largest eigenvalue of $\mathbf{T}$ must then be $\lambda_1 = 1$ with a corresponding eigenvector $\boldsymbol{v}_1$, describing the stationary distribution of the absorbing state. Interestingly, the second largest eigenvalue has been found to be close to unity $\lambda_2 \approx 1$ and much larger than the remaining $\mathrm{d}-2$ eigenvalues, for several simple walking systems [48.63]. This suggests the existence of a metastable neighborhood in state space captured by the distribution $\boldsymbol{v}_2$ with an associated time constant $\tau = -1/\log(\lambda_2)$. Other approaches to computing these stochastic stability bounds are also possible [48.64].

48.3.6 Input–Output Stability

Input-output methods define sensitivity measures to summarize the system's response to disturbances. For example, for a walking system, one could define a vector $\boldsymbol{l} \in \mathbb{R}^m$ composed of m gait variables that directly relate to failure modes (such as foot-ground clearance) and measure the effect that a set of input disturbances $\boldsymbol{w}$, have on their values. The *gait sensitivity norm* does exactly this [48.65]; here the dynamic system response $\|\partial \boldsymbol{l}/\partial \boldsymbol{w}\|_2$ is used as a measure of disturbance rejection. How is this response actually computed? In practice, computing the system response can be done experimentally by generating a variety of perturbations and measuring the effect in the gait variables. Although laborious, this approach is probably the most appropriate for physical systems. Alternatively, for simulated limit cycle walkers, linearization of the Poincaré return map and perturbation analysis yields a complete characterization of the dynamic system response.

More general tools from nonlinear control theory can also be applied. The L_2 gain is a sensitivity measure that has seen widespread application in nonlinear control theory. For complex dynamical systems such as walking robots, an upper bound on the L_2 gain can be approximated using storage functions, which can often be computed using convex optimization [48.66, 67].

48.3.7 Stability Margins

In practice, computing performance measures like the ones mentioned above for complex walking systems can be very difficult. As a result, researchers often employ heuristic stability margins to keep the robot away from the boundary of stability. Here the definition of stability can vary widely. The simplest stability margins rely only on the static configuration of the robot, where if the COM ground projection leaves the support polygon, this corresponds to an uncompensated moment on the foot resulting in a rotation along its edge. Popular static metrics include the distance between the ground projection of the COM and the edges of the support polygon or the minimum potential energy required to tip the robot over sideways [48.68]. Because these stability criteria are based only on statics, they are only practically useful for slow (or *quasi-static*) motions.

The most commonly used dynamic stability margin requires that the CoP (or ZMP) remains within the boundary of the support polygon. When this condition is satisfied, the foot cannot rotate around the boundary of the support polygon. Note that this is certainly very conservative – many bipedal robots walk dynamically even with point feet (resulting in a CoP that is *always* on the boundary of the support polygon) – but this metric has proven extremely useful for flat-foot dynamic walking. Goswami introduced the related concept of the foot rotation indicator (FRI) [48.69] that is defined as the point on the ground where the net ground reaction force would have to act to keep the foot from rotating; this point need not stay inside the support polygon and can therefore be applied for a wider range of gaits. However, the stability claims based on these criteria are valid only on flat terrain and they do not constitute necessary or sufficient conditions for stable walking in general [48.15, 55].

When the COM projection exits the support polygon, there are two ways to make it return: by accelerating the COM in the direction of the support polygon through changes in angular momentum or by changing the support polygon by taking a step. This basic insight led to stability margins based on *capture regions* [48.55, 60]. A capture region is simply the set of all capture points (defined in Sect. 48.3.4). The *capture margin* is a metric that determines the *capturability* of a walking system. If the capture region and support polygon overlap, it is defined as the maximum distance between the points in the capture region and the closest edge of the support polygon, otherwise it is the negative distance between the capture region and support polygon. Note that large values under this metric are related to the total capture region area, which is in turn related to the ability of the walking system to stabilize itself, e.g., with upper body motions that affect angular momentum [48.60]. Extensions to n-step capture margins have also been developed and described in detail [48.55].

48.4 Generation of Dynamic Walking and Running Motions

As mentioned in the brief historical introduction to this chapter, early legged robots were maintaining quasi-static equilibrium postures at all time. The transition to more dynamic motions took place at the beginning of the 1980s, associated with biped [48.4] and monopod [48.5] robots. As a result, this Section on dynamic legged locomotion will focus on results obtained mostly with biped robots, although most of the methods discussed here could be applied as well to other legged robots.

48.4.1 Early Offline Motion Generation Schemes

The very limited computing power of early robots allowed very limited online decisions. As a result, trajectories were usually computed offline, especially in complex situations such as for legged robots. Online motion generation has finally become possible only by the year 2000, thanks to the exponential increase in computing power and the continuous progress of numerical methods during the last decades.

Trajectory Optimization

One of the earliest approaches to computing walking and running motions was through numerical optimization [48.70]. The idea is to leverage the capacity of trajectory optimization methods to take into account nonlinear dynamics and objectives such as minimizing energy consumption, and compute the corresponding optimal motion. Unfortunately, the dynamics of legged robots are so complex that this approach was still stuck with simple planar models by the end of the 1990s [48.71–75]. It is only by the year 2000 that the first optimal walking motions for complete 3-D models could be computed [48.76, 77].

Fortunately, numerical methods have continued to improve (as discussed in Sect. 48.6.1), allowing researchers to solve complex problems such as optimizing trajectories and the mass distribution of the robot simultaneously and maximizing the open-loop stability of trajectories [48.78].

Unfortunately, this approach still requires time consuming computations that often cannot be realized online. Not being able to compute walking or running motions online limits the robots to a predefined set of precomputed actions, potentially ruining their versatility and reactivity. One way to alleviate this serious limitation is to generate a database of trajectories [48.79] that can be queried online, possibly conditioned on orders given to the robot [48.80] or the current state of the robot in order to improve its stability [48.40, 81, 82].

Artificial Synergy Synthesis and the ZMP Approach

Partitioning the problem is another early approach, that aimed at palliating the complexity of the dynamics of legged robots that hindered trajectory optimization. The idea is to assign some degrees of freedom of the robot to take care of dynamic constraints such as the ODI (48.13), allowing the rest of the robot to be operated more or less independently. This general approach has been called *artificial synergy synthesis* [48.10].

The original proposition in [48.10] was to use trunk rotations to ensure dynamic feasibility while the legs of the robot executed a given prerecorded motion. More precisely, leg motions and contact points $\boldsymbol{p}_i$ being predefined, the trajectory of the CoP z could be predefined accordingly, so that it was just a matter of solving the ordinary differential equation (ODE) (48.13) for this predefined z to obtain the required rotations of the trunk. This original proposition was demonstrated experimentally 20 years later on the Waseda University WL-12RV biped walking robot [48.83]. This method of predefining the trajectory of the CoP has eventually been called the *ZMP approach* to walking motion generation (remember the ZMP is just another name given to the CoP). This ZMP approach has been associated later with having feet always flat on the ground, excluding heel and toe rotation phases. Note, however, that such phases were considered in the original proposition [48.10].

Some aspects of this proposition can be questioned. It has been argued first of all that predefining the evolution of the CoP is not necessary nor even desirable [48.84, 85]. Then, the ODI (48.13) clearly shows that dynamic feasibility depends on both variations of the angular momentum $\mathbf{L}$, and motion of the COM $\boldsymbol{c}$ with respect to contact points, $\boldsymbol{p}_i$. While trunk rotations mostly involve variations of the angular momentum, a recent analysis showed that this has only a weak influence on the balance of legged robots [48.86]. Dynamic feasibility can be handled much more efficiently by adapting the motion of the COM, which has been the prevailing approach.

The core idea of partitioning the motion of the robot builds on a profound and far-reaching observation: if the robot has a sufficient number of degrees of freedom and sufficient control authority, then every part of the motion that is not involved in the dynamic constraints – everything but angular momentum and motion of the COM with respect to contact points – can be operated more or less independently, and appears therefore to be relatively peripheral to the problem of legged locomotion. It is the same key observation that implicitly drives the templates and anchors approach and the long history of simple biomechanical models of legged locomotion that focus on a few meaningful degrees of freedom, mostly the motion of the COM with respect to contact points, and abstract all the rest [48.87–89]. This idea has been tremendously successful in the legged locomotion research community.

48.4.2 Online Motion Generation: A Model Predictive Control Point of View

If the algorithms for motion generation could work sufficiently fast to be applied online, then the robot

could achieve reactivity and robustness by continuously adapting the motions to the current state of the robot and its environment. But therein lies a problem: how can we make sure that the continuous re-evaluation of online decisions maintains long term viability? One solution comes from the model predictive control (MPC) theory.

The online motion generation schemes that allow most of the great humanoid robots of today and yesterday to walk and run can all be related to MPC theory and can all be seen as variants of the same MPC scheme, although they have rarely been introduced in that way. Most adopt the artificial synergy synthesis approach described in the previous Section, focusing almost exclusively on the motion of the COM with respect to contact points and assuming that the rest of the robot can be operated more or less independently.

Viability, Optimal Control and MPC

When trying to identify viable states in Sect. 48.3.3, the first option was to check if the integral (48.32) could have a finite value. A classical result from optimal control theory is that an optimal control law minimizing this integral would make it decrease with time. As a result, if this integral was finite in the beginning, it would stay finite: if the state of the robot was viable in the beginning, it would stay viable. The problem is, the dynamics of legged robots is too complex to be able to compute such an optimal control law in the general case.

MPC is one way to obtain a computable approximation to this optimal control law. The fundamental idea is to introduce a *terminal constraint* [48.90], imposing that the integral (48.32) is equal to zero after a finite length of time. Introducing such an artificial constraint naturally yields a suboptimal control law, but this allows considering the integral (48.32) over only a finite length of time while preserving viability. Note that enforcing viability in finite time means in this case enforcing capturability: here, the terminal constraint is a capturability constraint.

Many MPC variants have been proposed [48.91], but two extremes are going to be significant in the following. One extreme is to observe that minimizing an integral of the form (48.32) is not necessary to obtain a viable behavior: the terminal, capturability constraint can be sufficient in this respect [48.92]. In this case, however, it is not possible to continuously re-evaluate and adapt the motion to the state of the system: this must be done at carefully chosen moments, what limited the reactivity of the robots. The other extreme is to observe that the terminal constraint is not absolutely necessary [48.93]: simply minimizing a truncated version of the integral (48.32) over a sufficiently long but finite length of time can still lead to an integral decreasing with time, ensuring viability. In this case, it is possible to continuously re-evaluate and adapt the motion to the state of the system without any problem.

Let us see now how all this theory is applied to the online generation of walking and running motions. In all the following examples, the robot is supposed to walk on a flat horizontal ground.

Predefined Footsteps and Capturability Constraint

Let us have a look first at motion generation schemes implementing the original ZMP approach, with predefined footsteps and a predefined CoP, and considering a capturability constraint to ensure viability. Let us begin with the walking motion generation scheme implemented in the long series of Waseda University humanoid robots [48.94], which considers a four point mass model with predefined motion except for the horizontal motion of the waist and trunk masses which is assigned to follow a reference trajectory for the CoP. An iterative procedure based on fast fourier transforms is used then to solve the dynamics

$$\frac{\sum m_i(\ddot{c}_i^z + g^z)\boldsymbol{c}_i^{x,y} - m_i\, c_i^z\, \ddot{\boldsymbol{c}}_i^{x,y}}{\sum m_i(\ddot{c}_i^z + g^z)} \longrightarrow \boldsymbol{z}_{\text{ref}}^{x,y} \,. \tag{48.39}$$

In order to execute this scheme online, a capturability constraint is introduced, imposing that the robot is always able to stop within two steps [48.95]. Unfortunately, details about the choice of the reference CoP and the exact terminal constraint are not disclosed.

The walking motion generation scheme implemented in the Munich University Johnny robot [48.96] considers only a three point mass model with predefined motion, except for the horizontal motion of the main mass in the trunk which is assigned to follow a piecewise linear reference trajectory for the CoP. One degree of freedom is left in this reference trajectory in order to impose a terminal constraint on the position of the COM at the end of the next two steps

$$\boldsymbol{c}^{x,y} = \boldsymbol{c}_{\text{ref}}^{x,y} \,. \tag{48.40}$$

This constraint appears to be incomplete in imposing capturability, what would require to consider also the velocity of the COM (Sect. 48.3.3).

The walking motion generation scheme implemented in the Honda Asimo robot [48.58] is very similar, three point masses and a piecewise linear reference for the CoP with one degree of freedom left to satisfy the terminal constraint. The difference lies in the terminal constraint which is a true capturability constraint, imposing cyclicity of the motion through the capture point/XCOM/divergent component of the dy-

namics at the end of the next step

$$\boldsymbol{\xi}^{x,y} = \boldsymbol{\xi}^{x,y}_{\text{ref}} \,. \tag{48.41}$$

The walking motion generation scheme implemented in the Tokyo University H7 robot [48.97] considers the whole dynamics (48.13) of the robot. The whole motion of the robot is predefined, except the horizontal motion of the COM which is assigned to follow a reference trajectory for the CoP, set in the middle of the contact points. An iterative procedure is used then to solve the dynamics

$$\boldsymbol{c}^{x,y} - \frac{m\,c^z\,\ddot{\boldsymbol{c}}^{x,y} - \mathbf{S}\dot{\mathbf{L}}^{x,y}}{m(\ddot{c}^z + g^z)} \longrightarrow \overline{\boldsymbol{p}_i^{x,y}} \,. \tag{48.42}$$

The terminal constraint (48.40) on the position of the COM (incomplete with respect to capturability) is also considered at the end of the next two steps.

The walking and running motion generation scheme implemented in the Toyota Partner robot [48.98] is exactly the same, except for the terminal constraint which is a true capturability constraint imposing cyclicity of the motion through both the position and velocity of the COM.

The walking and running motion generation scheme implemented in the Sony QRIO robot [48.99] follows a similar design, but considers only a single point mass at a constant height, with its horizontal motion assigned to minimizing the deviation of the CoP from the middle of the contact points

$$\min \int \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 \mathrm{d}t \,, \tag{48.43}$$

while imposing a capturability constraint on both the position and velocity of the COM.

Another variant tested on the Kawada HRP-2 robot [48.100] also considers a single point mass at a constant height, but the CoP follows a piecewise polynomial trajectory

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} = \boldsymbol{z}^{x,y}_{\text{ref}} \,, \tag{48.44}$$

with some degrees of freedom left to satisfy the same capturability constraint as before, through both the position and velocity of the COM. An important characteristic of this scheme is that the piecewise polynomial trajectory of the CoP may fluctuate strongly, threatening to violate the ODI (48.13). An automatic adjustment of the step timings is proposed therefore in order to minimize this risk.

All of these walking and running motion generation schemes try to impose capturability through terminal constraints, but some of them appear to fail properly doing so by only constraining the position of the COM. None of them consider an integral of the form (48.32): the integral (48.43) does not match. We have seen earlier that in this case, it is possible to re-evaluate and adapt the motion at specific instants, what has been done when having to realize new steps, or when changing the walking speed or direction. But to adapt the motion to a perturbation, a specific observer would be required to trigger the adaptation at the correct instant [48.92], and it appears that this option hasn't been investigated: no state feedback has been experimented with these motion generation schemes.

Predefined Footsteps, Without Capturability Constraint

In the standard walking motion generation scheme implemented in the Kawada HRP-2 humanoid robot [48.101], the whole motion of the robot is predefined as before, except the horizontal motion of the COM which is assigned this time to minimize the weighted integral

$$\min \int \|\dddot{\boldsymbol{c}}^{x,y}\|^2 + \beta \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} + \frac{\mathbf{S}\dot{\mathbf{L}}^{x,y}}{mg^z} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 \mathrm{d}t \tag{48.45}$$

of the norm of the third derivative of the motion of the COM and the deviation of the CoP from a reference in the middle of the contact points. This is clearly an integral of the form (48.32), and we have seen that viability can be ensured in this case if we consider a truncated version of this integral over a sufficiently long but finite length of time, typically the next two steps, without the need to impose any terminal constraint. We have also seen that it is possible in this case to continuously re-evaluate and adapt the motion to the state of the system, a clear improvement over the previous approaches based solely on terminal constraints. This has been validated experimentally in various situations, effectively adapting the walking motion to perturbations [48.102].

An interesting variant [48.103] introduces variations of the angular momentum L as an additional variable to minimize the combined integral

$$\min \int \|\dddot{\boldsymbol{c}}^{x,y}\|^2 + \beta \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} + \frac{\mathbf{S}\dot{\mathbf{L}}^{x,y}}{mg^z} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 + \gamma \|\boldsymbol{L}^{x,y}\|^2 \mathrm{d}t \,, \tag{48.46}$$

showing an improvement in the tracking of the generated motion (and the same closed loop robustness to small perturbations).

Avidbots Ex1016
Page 1243 of 2154

A problem however with these approaches is that simply minimizing the deviation of the CoP from the middle of the contact points doesn't preclude it from fluctuating, so the ODI (48.13) could be violated, especially in case of perturbations. This is monitored in [48.102] to trigger a change of footstep if necessary, but without any clear guarantee that this change is appropriate. For this reason, it has been proposed in [48.85] to impose the ODI (48.13) as a strict constraint, considering a single point mass model with a constant height

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} , \tag{48.47}$$

and simply minimize the integral

$$\min \int \|\ddot{\boldsymbol{c}}^{x,y}\|^2 \mathrm{d}t \tag{48.48}$$

over a sufficiently long but finite length of time to ensure viability. This is the walking motion generation scheme implemented in the Aldebaran Nao robot [48.104].

A variant tested successfully on the DLR (Deutsches Zentrum für Luft- und Raumfahrt) biped robot [48.105] considers minimizing the deviation from a reference trajectory for the XCOM together with the derivative of the CoP

$$\min \int \left\|\boldsymbol{\xi}^{x,y} - \boldsymbol{\xi}_{\text{ref}}^{x,y}\right\|^2 + \beta\|\dot{\boldsymbol{z}}^{x,y}\|^2 \mathrm{d}t \, . \tag{48.49}$$

An additional terminal constraint was considered but not tested.

A last proposition, tested only in simple simulations, not with a real robot [48.106], considers a single point mass which is assigned to minimize the deviation of the leg length l from a given reference, and the deviation of the CoP from the middle of the contact points over the next two steps

$$\min \int (l - l_{\text{ref}})^2 + \beta \left\| \boldsymbol{c}^{x,y} - \frac{c^z\,\ddot{\boldsymbol{c}}^{x,y}}{\ddot{c}^z + g^z} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 \mathrm{d}t \, . \tag{48.50}$$

What makes this proposition particularly stimulating is that this is not an integral of the form (48.32), that can ensure viability, and no terminal constraint is there to impose capturability, so there is no direct relation with the viability/capturability analysis developed in Sect. 48.3.3, on which all the previous schemes were based. What this minimization only imposes is the ability to realize two more steps in the future, with legs approximately at a nominal length: the situation in the end of these two steps is not controlled (no terminal constraint), so the robot might very well be falling afterwards. However, in the presented simulations, it appears that maintaining this ability to make two more steps in the future is a sufficient condition for generating stable walking and running motions. The conclusion is that terminal constraints or integrals of the form (48.32) are in fact not mandatory and could be relaxed altogether.

In all of these approaches, the footsteps were predefined and kept fixed, which is obviously a strong limitation on the robot's capacity to adapt to a changing environment or to strong perturbations.

Adaptive Footsteps

In fact, adapting foot placement is straightforward, and has already been validated on a Kawada HRP-2 robot [48.107]: the only required change is to consider foot placement as a decision variable, used in addition to the horizontal motion of the COM, in order to both satisfy the ODI (48.47) and minimize the integral

$$\min \int \left\|\dot{\boldsymbol{c}}^{x,y} - \dot{\boldsymbol{c}}_{\text{ref}}^{x,y}\right\|^2 \mathrm{d}t \tag{48.51}$$

over a sufficiently long length of time to ensure viability (since this is clearly an integral of the form (48.32)), and have the COM follow on top of that the reference velocity $\dot{\boldsymbol{c}}_{\text{ref}}^{x,y}$. Now, since the footstep placement is decided online, geometric feasibility needs to be checked online as well. A simple but effective option is to consider a polygonal approximation of the reachable volume of the COM with respect to each foot on the ground [48.108].

Yet, the timing of the steps was still predefined, what has a huge influence on the reactivity of legged robots [48.60]. In order to adapt it as well, it has been proposed in [48.109] to minimize the combined integral

$$\min \int \left\|\dot{\boldsymbol{c}}^{x,y} - \dot{\boldsymbol{c}}_{\text{ref}}^{x,y}\right\|^2 + \|\ddot{\boldsymbol{f}}^{x,y}\|^2 \mathrm{d}t \, , \tag{48.52}$$

with $\ddot{\boldsymbol{f}}^{x,y}$ the horizontal acceleration of the feet. But this approach has been tested only in simple simulations so far. Moreover, the underlying optimization problem becomes nonlinear, what is significantly more involved to solve.

Another approach, quite unique, starts with a singular linear quadratic regulator (LQR) design [48.110], considering the classical single point mass at a constant height, assigned to minimize the deviation of the CoP from a combination of a reference CoP and the XCOM

$$\min \int \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} + \alpha \boldsymbol{z}_{\text{ref}}^{x,y} - (1+\alpha)\boldsymbol{\xi}^{x,y} \right\|^2 \mathrm{d}t \tag{48.53}$$

for some $\alpha > 0$. An interesting feature of this singular LQR design is that it can be solved analytically. And an

interesting property of this singular objective function is that it can be reduced to zero, in which case we have

$$c^{x,y} - \frac{c^z}{g^z}\ddot{c}^{x,y} = (1+\alpha)\xi^{x,y} - \alpha z_{\text{ref}}^{x,y} , \tag{48.54}$$

what can be combined with the dynamics (48.35) of the XCOM to obtain a stable first order dynamics

$$\dot{\xi}^{x,y} = \alpha\omega(z_{\text{ref}}^{x,y} - \xi^{x,y}) , \tag{48.55}$$

according to which the XCOM ξ is going to converge to $z_{\text{ref}}^{x,y}$, and the CoP z as well according to (48.54). But in a very unusual twist, this LQR design is not used as is, and is inverted analytically to find under which condition does a piecewise constant trajectory of the CoP generate a nondiverging motion of the COM. Unsurprisingly, this nondiverging condition ends up being a terminal constraint on the capture point/XCOM/diverging part of the dynamics, with trajectories corresponding in the end exactly to those found in [48.59]. But here, this terminal constraint is used in the end to decide online the footstep placement that will ensure viability, what is eventually validated experimentally with very strong perturbations.

All Variants of a Common Design

The viability analysis of Sect. 48.3.3 attested the crucial importance of anticipation to avoid falling. We observe now that the walking and running motion generation schemes that power most of the greatest humanoid robots of today and yesterday, the Honda Asimo, the Toyota Partner, the Sony Qrio, the Aldebaran Nao, the Kawada HRP-2, the Tokyo University H7, the Munich University Johnny, the DLR biped, the long series of Waseda University humanoids and many others, are all structured around this necessity to anticipate, at least a couple steps ahead of time. Viability is secured then in different ways, always related to MPC theory, through capturability or through minimizing an integral of the form (48.32). In all cases, adopting the artificial synergy synthesis approach, focusing on the motion of the COM of the robot with respect to contact points, considering that the rest of the motion can be handled more or less independently, is the key to enable efficient computations. All these motion generation schemes appear then to share the same general design: MPC of the COM of the robot with respect to contact points. The rest is details.

The compelling consequence of identifying this common design, is revealing the possibility to cross-breed all these approaches, retaining the best features of each: more precise multiple mass models, for generating both walking and running motions, with adaptive footstep placement and adaptive timing, sensor feedback, all in order to obtain in the end the ultimate robust and versatile online motion generation scheme. These are the next obvious steps.

Missing in this gallery of great legged robots, the Boston Dynamics biped and quadruped robots. They are without a doubt the legged robots exhibiting the most impressive dynamic motions today, robust and versatile. Exact details about their control algorithms are scarce, but various clues suggest that they might share the same design.

48.4.3 Motion in Constrained Environments

We have seen so far how to generate dynamic walking and running motions by considering the dynamic feasibility constraint (48.13) independently from any other concern. But on truly rough terrain or in cluttered spaces, kinematic constraints on the motion of the robot cannot be disregarded and can further complicate the problem. More precisely, kinematic constraints generally make the reachable space of the robot nonconvex, so that deciding actions and motions with only a local view on the situation can quickly get the robot stuck in local loops or dead-ends. One way to decide actions and motions with a global view on the situation is with planning techniques, as described in Chap. 7 on motion planning and Chap. 47 on motion planning and obstacle avoidance. In the case of legged robots, three classes of problems of increasing complexity have been considered: dynamic manipulation without locomotion, locomotion on a flat ground with obstacles, and locomotion on rough terrain with complex contact transitions.

Dynamic Manipulation Without Locomotion

In the case of manipulation tasks without locomotion, the only difference of legged robots with respect to more traditional manipulator robots is the dynamic feasibility constraint (48.13). Therefore, a mere application of the standard planning techniques described in Chaps. 7 and 47 including this feasibility constraint has proved to be fully sufficient. But these techniques are strongly biased towards quasi-static motions. As a result, a sequence of statically stable postures is computed first, and the motion is smoothed and accelerated afterwards, finally taking into account dynamic feasibility [48.111]. Much more involved are the problems involving locomotion, which attracted therefore much more attention.

Legged Locomotion on a Flat Ground with Obstacles

If we consider that the problem of generating dynamic legged locomotion is properly solved thanks to the dif-

Avidbots Ex1016
Page 1245 of 2154

ferent methods described earlier in this section, we can try to abstract its details and plan the navigation of the upper body of a legged robot between obstacles as for any other mobile robot, with the same planning techniques, regardless of the exact locomotion method. It has been shown indeed that legged robots present a form of *small-space controllability*, which means that their upper-body can follow any given path with any given accuracy [48.112], although this can require impractically quick steps. A different option to follow exactly any given path with the upper-body is to cross legs, as if walking on a balance beam, and some robots have been specifically designed to be able to realize such feats [48.113], but this can require impractically slow steps. Ruling out impractically slow or impractically quick steps, the legged locomotion may finally differ from the planned path and fail to avoid obstacles. In this case, iterative replanning techniques have been proposed [48.114].

In this approach, however, it is assumed that finally finding proper footholds to follow the path planned for the upper body is not a problem. Yet, the appeal of legged robots is to provide improved mobility on difficult terrain, situations where finding footholds is actually a problem. A solution then is to check the availability of proper footholds when planning the upper body motion [48.115], an approach recently refined to show how to obtain in the general case an exact equivalence between planning a continuous path and planning a sequence of footholds on a flat ground [48.116, 117].

A different approach is to plan directly the footholds with respect to obstacles on the ground, and generate afterwards the corresponding dynamic legged locomotion [48.118]. This has been applied successfully to long distance navigation with a tiered strategy [48.119], with computations fast enough to run online in changing environments [48.120], taking into account deterministically moving obstacles [48.121], on mildly rough terrain [48.122–124], using on-board sensing [48.125]. Obstacles above the ground can be considered then with a hybrid bounding box or more refined swept volumes [48.126, 127], or one can approximate the free-space with convex segmentation [48.128].

Legged Locomotion with Complex Contact Transitions

On truly rough terrain or in cluttered environments, legged locomotion may require much more complex contact situations and contact transitions than in standard walking or running, involving not only contacts between the feet and the ground, but all potential contact surfaces on the robot and on the environment. The variability of these situations is beyond comparison with the previous case on flat ground, and chances of precomputations or simplifications are strongly lacking in respect. Notably, the kinematic and dynamic constraints cannot be tackled independently, and having to consider them simultaneously results in an overwhelming computational problem. As a result, a major effort has been to find a proper ordering of the different subproblems in order to obtain a tractable problem in the end, finally opting for planning contacts before motion [48.129–131]. Complex real-life use-cases have been solved this way [48.132–134], but the computational requirements still limit our ability to achieve the versatility of locomotion possible on flat ground. Online adaptation to changing environments has yet to be convincingly demonstrated and remains an active area of research.

As noted before, the underlying planning techniques manifest a strong bias towards quasi-static motions: some post-processing is necessary to obtain a truly dynamic motion in the end. The same two options as in Sect. 48.4.1 have been proposed, standard trajectory optimization [48.135], or following the artificial synergy synthesis approach and computing first the motion of the COM of the robot with respect to the different contact points, independently from the rest of the motion of the robot, which is computed only afterwards [48.21]. But the strong interdependency between kinematic and dynamic constraints defeats the core assumption of the artificial synergy synthesis approach in this case: dynamic feasibility can't always be handled independently from the rest of the motion of the robot, so there is a significant risk that this method fails finding in the end a feasible motion even if one exists.

Further discussion of all these problems in the specific case of humanoid robots can be found in Chap. 67.5 on whole-body activities of humanoid robots, or in [48.136]. There is also a rich literature on gait selection for many-legged robots, starting as early as [48.137] but this is still an active area of research [48.138].

48.4.4 Motion Generation with Limited Computing Resources

The walking and running motion generation schemes presented so far in this chapter all require significant computation, even when they consider only simplified dynamical models. Early legged robots, which had very limited computing resources, had to rely on much less demanding approaches. One option is to combine simple explicit rules, each one focusing on a different part of the overall motion. Another option is to try to mimic biological hypotheses on how animals generate and control their motion, what ended up in fact with somehow similar propositions.

Combining Simple Rules

Thanks to simplifying symmetries in biped or quadruped running gaits such as trot, pace and bound, the whole family of robots hopping in the MIT LegLab throughout the 1980s on one, two or four legs, in 2-D (two-dimensional) or 3-D (three-dimensional), could rely on the same simple control design [48.5]. The general idea is to apply simple control laws independently to the different parts of the overall locomotion task. The vertical oscillations were controlled by open-loop vertical thrusts, simply stabilized by mechanical energy losses. Since the angular momentum is conserved during flight phases, the body attitude was controlled only during stance phases, with standard proportional–derivative (PD) control. The key to maintaining long-term balance, and controlling the global motion was foot placement. An elementary analysis of the motion of the COM during the stance phase revealed the role of the so-called *neutral* foot position, which would make the stance phase perfectly symmetric. It is then just a matter of linear corrections of the foot position with respect to this neutral position to make the stance phase asymmetric in order to control the robot's speed and balance. The direct combination of these utterly simple control laws resulted in impressively robust and versatile locomotion.

The first biped robot that walked dynamically relied on an equally simple approach [48.4]. This robot emulated walking on stilts, with contact surfaces between the feet and the ground reduced to single points, so $z = p$. With a single, fixed contact point during single support phases (the double support phases were supposed negligible), the ODI (48.29) can be solved analytically: for $t \geq t_0$,

$$\begin{pmatrix} c^{x,y}(t) - p^{x,y}(t_0) \\ \dot{c}^{x,y}(t) \end{pmatrix} = A(t) \begin{pmatrix} c^{x,y}(t_0) - p^{x,y}(t_0) \\ \dot{c}^{x,y}(t_0) \end{pmatrix} \tag{48.56}$$

with

$$A(t) = \begin{pmatrix} \cosh \omega(t-t_0) & \omega^{-1} \sinh \omega(t-t_0) \\ \omega \sinh \omega(t-t_0) & \cosh \omega(t-t_0) \end{pmatrix},$$
$$\omega = \sqrt{\frac{g^z}{c^z}}\,. \tag{48.57}$$

If the next single support phase starts at time t_1 with a foot position $p^{x,y}(t_1)$, we end up having

$$\begin{pmatrix} c^{x,y}(t_1) - p^{x,y}(t_1) \\ \dot{c}^{x,y}(t_1) \end{pmatrix} = A(t_1) \begin{pmatrix} c^{x,y}(t_0) - p^{x,y}(t_0) \\ \dot{c}^{x,y}(t_0) \end{pmatrix} + \begin{pmatrix} p^{x,y}(t_0) - p^{x,y}(t_1) \\ 0 \end{pmatrix}, \tag{48.58}$$

which describes the motion of the COM $c^{x,y}$ with respect to the contact points $p^{x,y}$, from one step, starting at time t_k, to the next, starting at time t_{k+1}. This discrete time dynamical system can be controlled then with the foot placement $p^{x,y}(t_k)$. Since it is linear, standard pole placement can be applied, yielding a standard PD control of foot placement. This simple approach has been successfully applied on more complex biped robots [48.139, 140].

The drawback of combining independent control laws is to lack coordination yielding potentially harsh, unrefined motion, but this approach demonstrates that stable locomotion is possible even with very limited control resources. Of primary importance here are foot placement and upper body attitude; the rest of the motion appears to be secondary, at least with respect to balance control. Pushing this observation further, it has been proposed, and validated in simulation, that a large variety of walking patterns could be used, and stabilized only with upper body attitude and foot placement control, both with simple PD control laws [48.141]. It is not surprising then that hand-designed motion patterns can be used successfully on robots as complex as the Korea Advanced Institute of Science and Technology (KAIST) Hubo humanoid robot [48.142], although this may require delicate fine tuning.

Reducing even further the control requirements, the mechanics of the robots could be tuned so that passive motions would automatically land the feet on the appropriate locations with respect to stability, ending up with perfectly passive stable dynamic locomotion. This idea will be discussed more thoroughly in Sect. 48.6.

Biomimetic Motion Generation

Current theories on locomotion control in animals include central pattern generators (CPGs) and cascades of reflex motions, which interact and combine to generate the final motion. CPGs are tunable oscillators, generating synchronized quasi-cyclic motion patterns in response to simple control signals such as locomotion speed or turning angle [48.143]. In order to introduce such CPGs in legged robots, standard oscillators such as the Van der Pol equation or the Hopf oscillator have been proposed [48.144, 145], or more biologically inspired neural oscillators [48.146] (ironically, *central* pattern generators often happen to be *decentralized*, composed of a collection of synchronized oscillators [48.143, 145, 147]).

But CPGs are intrinsically open-loop motion generators, so to stabilize and adapt the motion of the robot to their environment, limited feedback loops have been introduced, similar to the simple control laws discussed earlier in this Section, focusing on upper body attitude and foot placement [48.145, 147, 148]. Contin-

uing with the biological analogy, these simple feedback loops have been called reflexes. Following ideas discussed more thoroughly in Chap. 13 on behavior-based systems, the *subsumption architecture* has been proposed to build much more complex networks of reactive behaviors or reflexes [48.149], and has allowed a hexapod robot to walk successfully on mildly rough terrain without any careful planning of the motion [48.150].

Fundamentally the Same Decoupling Approach

Being inspired by simple mechanical analyses or by animals, these approaches based on combinations of simple rules demonstrate that stable walking can be realized without much computation, at least in simple situations. More complex situations may require more refined motions and therefore more refined motion generation schemes, such as the ones discussed earlier in this chapter. But it is striking to observe again the same focus here on the motion of the COM of the robot with respect to the contact points, when controlling either the upper body attitude or the foot placement (or both), regardless of the rest of the motion, which can be generated by CPGs, inverse kinematics or any other approach, simple or complex. This is fundamentally the same decoupling as proposed earlier with the artificial synergy synthesis approach.

48.5 Motion and Force Control

Properly executing the walking and running motions computed in the previous section requires precise control of the motion of the robot and of the contact forces with the environment, applying techniques from Chapters 8 and 9 on motion and force control. But since legged robots are often complex mechanical systems with many degrees of freedom and some form of redundancy, techniques from Chap. 10 on redundant manipulators also need to be applied frequently. And in case of contact with multiple surfaces, problems of contact force distribution appear, requiring techniques similar to those found in Chap. 38 on grasping. And in the opposite case, during flight phases, when the robot is not at all in contact with its environment, its dynamics appear to share important properties with free-floating space robots, requiring control techniques similar to those found in Chap. 55 on space robotics. All these techniques (and many more) are necessary to make legged robots walk and run efficiently. The purpose of this Section is to see how they connect and are implemented in the case of legged robots.

Whole-Body Motion

Humanoid and other legged robots are often complex mechanical systems with a large number of degrees of freedom in order to accomplish the various kinematic and dynamic tasks necessary for simultaneously achieving locomotion, perception and manipulation of the environment, interaction with humans, etc. Making use of all available degrees of freedom, and not only those in the legs, to achieve multiple simultaneous objectives, and not only locomotion, is usually called *whole-body motion control* (Chap. 67.5). We have seen previously that locomotion mostly involves the motion of the COM $\boldsymbol{c}$ with respect to points $\boldsymbol{p}_i$ of contact with the environment. Other objectives involve end-effector motion for manipulation, gaze control for perception, motion of specific parts of the robot for obstacle avoidance, etc. In all of these cases, what needs to be controlled is Cartesian motion of different parts of the robot, rather than joint motion. Several control schemes have been proposed to do this, standard inverse kinematics followed by joint space control [48.139], virtual model control [48.151], the task function approach [48.152, 153], operational space control [48.154].

Interestingly, the last two options allow specifying different priorities to the different objectives assigned to the robot, so in case these objectives are not feasible jointly, the robot tries first to achieve the objectives with higher priority. In the case of legged robots, maintaining balance and avoiding falls appear to be of primary importance for the safety of both the robot and its environment, so being able to consider this objective with a higher priority can be crucial in this regard. These control schemes allow moreover some form of decoupling between the different objectives, further contributing to the artificial synergy synthesis approach introduced earlier in the previous Section. We are going therefore to continue focusing here exclusively on the locomotion objective, since the other ones can be considered more or less independently and are not specific to legged robots.

Flight Phases

We have seen earlier in this chapter that during flight phases, when a legged robot is not in contact with its environment, the motion of the COM invariably follows a standard falling motion: there is no possibility to make it move in any other way. But we have seen that even though the angular momentum $\mathbf{L}$ is constant during flight phases, the joint configuration $\hat{\boldsymbol{q}}$ and the global orientation $\boldsymbol{\theta}_0$ of the robot can still be driven

Avidbots Ex1016
Page 1248 of 2154

concurrently together to any desired value thanks to the nonholonomy of the angular momentum (48.5). However, a famous theorem due to *Brockett* [48.155] implies that continuous time-invariant feedback control laws such as those proposed in [48.156, 157] are not able to do so. The solution is to use discontinuous or time-varying control laws [48.158], an approach well established in space robotics [48.159, 160].

In the case of running motions, it has been proposed to specifically adapt leg trajectories during flight phases in order to control both the joint configuration and the global orientation at the end of the flight phases [48.161]. It has been measured indeed that a simple difference in the height of a single step can result in 1 or 2 degrees of difference in the global orientation of the robot [48.8]. Albeit small, this effect is far from negligible. More thorough discussions of nonholonomic constraints and the impact of Brockett's theorem can be found in Chap. 49 on the modeling and control of wheeled mobile robots or in [48.158].

Angular Momentum

Either considering the ODI (48.13) when walking on a flat ground, or the Newton and Euler equations (48.3) and (48.4) in the general case, the dynamic feasibility of legged motions appears to be directly and entirely related to the motion of the COM $\boldsymbol{c}$, and the angular momentum, $\mathbf{L}$. It has been naturally proposed therefore to control specifically those two quantities concurrently [48.162–164]. Of course, since COM motion and angular momentum are directly related to contact forces $\boldsymbol{f}_i$ (or equivalently to the CoP, z), controlling the former is absolutely equivalent to controlling the latter. The question however is whether the angular momentum should be specifically controlled to some given value and what that value should be [48.163, 164].

Of course, since the angular momentum is directly related to contact forces, it should comply with the unilaterality and Coulomb friction limitations on contact forces (48.11) and (48.17). As a matter of fact, the regulation of angular momentum has been discussed only in quasi-static situations [48.162–164], where the reference value is clearly zero, which generally complies well with these limitations. And in the only case where walking was considered, the horizontal angular momentum $\mathbf{L}^{x,y}$, that appears in the ODI (48.13) when walking on a flat horizontal ground, was explicitly not controlled [48.162]. It appears, in fact, that during walking motions, the question of which value the angular momentum should have is very intricate, because of its nonholonomy, but it is definitely not zero [48.8, 103].

Even in quasi-static situations, both joint configuration and global orientation can be controlled independently from the angular momentum; the connection between these different values is very subtle. As a result, controlling the angular momentum does not induce a control of either joint configuration or global orientation (due to its nonholonomy). The opposite is true however: controlling joint configuration and global orientation does induce a control of the angular momentum as a derived quantity. It has been argued furthermore that common whole body motion objectives such as manipulation or interaction with humans are more directly related to joint configuration and global orientation than to angular momentum [48.165]. As a matter of fact, disregarding the exact variations of the angular momentum has been the most frequent option so far [48.139, 151–154, 165]. In the end, it is not clear if specifically controlling the angular momentum can help controlling legged robots: this is still an open question.

COM Motion Control

Putting the angular momentum aside, and considering only the simple linear dynamics (48.29) when walking on a flat ground, and more precisely its decomposition (48.34) and (48.35), we have seen that every state satisfying the condition (48.36), such that the capture point $\boldsymbol{\xi}$ is in the support polygon, is capturable and can therefore be stabilized. If the capture point is strictly inside the support polygon, the COM can even be stabilized to any reference position $\boldsymbol{c}_{\text{ref}}$ above the support polygon, with a simple linear feedback of the capture point

$$z^{x,y} = c^{x,y}_{\text{ref}} + k(\xi^{x,y} - c^{x,y}_{\text{ref}}) \,, \tag{48.59}$$

where the feedback gain $k > 1$ must be tuned in order to keep the CoP z within the support polygon, ensuring dynamic feasibility of the motion. Indeed, using this feedback with the dynamics (48.35) yields a stable closed loop dynamics

$$\dot{\xi}^{x,y} = \omega(k-1)(c^{x,y}_{\text{ref}} - \xi^{x,y}) \,, \tag{48.60}$$

according to which the capture point $\boldsymbol{\xi}$ will converge to the reference position $\boldsymbol{c}_{\text{ref}}$, and the COM $\boldsymbol{c}$ will converge to it as a result of the stable dynamics (48.34). It has been shown that among all proportional-derivative (PD) feedback of the motion of the COM

$$z^{x,y} = c^{x,y}_{\text{ref}} + k(c^{x,y} + \lambda\dot{c}^{x,y} - c^{x,y}_{\text{ref}}) \,, \tag{48.61}$$

the feedback (48.59) of the capture point, with a choice of $\lambda = \omega^{-1}$, is the only one that will successfully stabilize all capturable states [48.166]. It has been successfully applied to walking motions on the DLR biped robot [48.59].

Now, if there is a time-varying perturbation or error $\boldsymbol{\varepsilon}$ in our dynamics, such that

$$c^{x,y} - \frac{c^z}{g^z}\ddot{c}^{x,y} = z^{x,y} + \varepsilon \,, \tag{48.62}$$

the PD feedback (48.61) can be shown to be robustly stable (input-to-state stable, to be precise) with respect to $\boldsymbol{\varepsilon}$, but with $\lambda < \omega^{-1}$ and $k > \frac{1}{\lambda^2\omega^2}$ [48.167]. As discussed above, with this choice of λ, not all capturable states will be stabilized successfully, even if there is no perturbation. This is unfortunately a usual trade-off in robust control.

Force Feedback

Motion control of legged robots without contact force sensing and feedback has been shown to be possible for simple quasi-static balancing tasks, using passivity based control [48.168]. However, force sensing and feedback is commonly used when realizing more dynamic motions such as walking or running. One of the first problems to overcome in this case is the compliance of both the robot and its environment, which easily generates destabilizing oscillations [48.169–172]. Contact force control schemes of various complexity have been proposed, but always with the goal of damping these undesired oscillations. The exact implementation of these force control schemes generally relies on a stiff lower-level joint motion control in order to enhance disturbance rejection, leading to admittance control implementations [48.139, 169–173]. More details on such control schemes can be found in Chap. 9 on force control.

Now, these damping control schemes inevitably introduce some delay in the realization of the desired contact forces. One option is to show that the COM motion control scheme described earlier, which entirely relies on contact forces, is robust to such delay [48.59]. Another option is to take this delay into account with a simple first-order dynamics of the CoP

$$\dot{z} = \omega_z(z_d - z) \,, \tag{48.63}$$

and adapt the previous COM motion control scheme accordingly [48.174, 175]

$$z_d^{x,y} = c_{\text{ref}}^{x,y} + k(\xi^{x,y} - c_{\text{ref}}^{x,y}) + k'(z^{x,y} - c_{\text{ref}}^{x,y}) \,. \tag{48.64}$$

This introduces some force feedback directly in the COM motion control through the measurement of the CoP z, what can help improve the reactivity of the controller to perturbations.

Contact Forces Distribution

In case of contact with multiple surfaces, computing contact forces $\boldsymbol{f}_i$, that allow realizing the desired motion control scheme through the Newton and Euler equations (48.3) and (48.4), and yet comply with both the unilaterality condition (48.11) and the Coulomb friction model (48.17), can be slightly more involved. The standard approach is through optimization, and more precisely quadratic programming (QP) with linear constraints, using a faceted, linear approximation of the Coulomb friction model (48.17)

$$\boldsymbol{A}\boldsymbol{f}_i \leq \boldsymbol{b} \,. \tag{48.65}$$

It has been proposed then to minimize when possible the norm of the joint torques $\boldsymbol{u}$ in order to reduce energy consumption [48.152],

$$\min_{\boldsymbol{f}_i} \ \|\boldsymbol{u}\|^2 \,, \tag{48.66}$$

or to balance the contact forces between the different contact points, either by minimizing the sum of their norms [48.165],

$$\min_{\boldsymbol{f}_i} \ \sum_i \|\boldsymbol{f}_i\|^2 \,, \tag{48.67}$$

or the norm of their difference [48.176],

$$\min_{\boldsymbol{f}_i} \ \|\boldsymbol{f}_{\text{left}} - \boldsymbol{f}_{\text{right}}\|^2 \,, \tag{48.68}$$

or minimizing the tangential forces in order to reduce the risk of slipping [48.177],

$$\min_{\boldsymbol{f}_i} \ \sum_i \|\boldsymbol{f}_i^{x,y}\|^2 \,. \tag{48.69}$$

More general QP formulations have been proposed [48.178, 179], including hierarchies of kinematic and dynamic tasks and constraints [48.153].

But contact forces obtained in these ways are not explicitly kept away from the edges of the friction cone and unilaterality condition, so that the slightest perturbation may quickly lead to sliding or tipping, what can lead to serious issues. The simplest way to steadily improve the robustness of the robot to such perturbations is to introduce a security margin w

$$\boldsymbol{A}\boldsymbol{f}_i \leq \boldsymbol{b} - w\boldsymbol{1} \,, \tag{48.70}$$

where $\boldsymbol{1}$ is a vector of ones, but there is no obvious way to decide which value would be appropriate. One option then is to try to maximize it with a linear program (LP)

$$\max_{w,\boldsymbol{f}_i} \ w \,, \tag{48.71}$$

improving as much as possible the robustness of the robot to perturbations by selecting contact forces $\boldsymbol{f}_i$ the farthest possible from the edges of the (faceted) friction cone and unilaterality condition. Note that the maximal value w^* obtained in this way is the *residual radius* discussed in [48.19].

48.6 Towards More Efficient Walking

Examples of remarkably efficient locomotion are ubiquitous in nature. From sprinting cheetahs to leaping sifakas, the variety and beauty of animal motion has inspired roboticists for decades. Yet some of the most impressive examples of humanoid walking, e.g., Honda's Asimo or Boston Dynamics' Atlas, have employed high-impedance joint control to achieve stable locomotion. As a result, the fluidity of their motions is far from that of their biological counterpart. Indeed, these robots expend an order of magnitude more energy than a typical human while walking [48.180].

In contrast, the passive dynamic walking machines of McGeer and others [48.6, 181] have been able to achieve surprisingly human-like gaits without any actuation, relying solely upon their carefully designed body dynamics. These machines moved down sloped terrain, relying upon the conversion of potential to kinetic energy to balance the losses introduced by impacts and friction. Subsequent examples of minimally-actuated bipeds capable of walking under their own power on flat terrain made the first steps toward realizing efficient walking machines [48.180, 182].

The existence of these robots leads to several interesting questions. For example, how can we find limit cycles of underactuated walkers and reason about their stability? Simple models of walking systems have been used to try to answer these questions for over two decades. Such analyses typically rely upon linearization of the model dynamics to integrate the equations of motion or numerical simulation of the nonlinear dynamics and analysis via Poincaré maps (Sect. 48.3.2). For example, the compass gait model is one of the simplest walking models, consisting of three point masses – one at the hip and one on each leg – and the legs connected to the hip with a pin joint (Fig. 48.10). Although even this very simple model cannot be analyzed analytically, a variety of stable gaits have been identified using numerical techniques [48.186].

Today there exists a community of researchers working to bridge the gap between stable and versatile machines like Asimo and efficient dynamic walkers. Topics of particular interest include navigating robustly over rough terrain, generating and stabilizing nonperiodic gaits, and scaling up to more complex robot models. New computational tools for generating and stabilizing trajectories for underactuated dynamical systems have been an important result of this work. In this section we highlight the fundamental insights provided by work on passive dynamic walking machines and describe ways in which these ideas have been extended to more general systems.

48.6.1 Gait Generation for Dynamic Walking

The most direct approach to producing efficient walking gaits in underactuated robots is to design controllers that produce actuated trajectories that mimic passive ones. Fundamental to these methods is the idea of restoration of mechanical energy. Since mechanical energy is

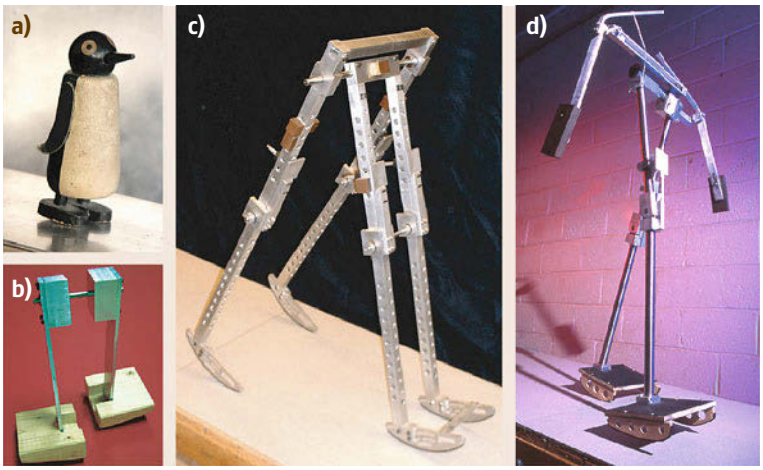


Fig.48.9a–f Passive-dynamic walking machines. **(a)** The Wilson *Walkie* (after [48.183]). **(b)** A slightly improved version (after [48.184]). Both **(c)** and **(d)** walk down a slight ramp with the *comical, awkward, waddling gait of the penguin* (after [48.183]). **(e)** Cornell copy (after [48.185]) of McGeer's capstone design. This four-legged *biped* has two pairs of legs, an inner and outer pair, to prevent falling sideways. **(f)** The Cornell passive biped with arms (photo by Hank Morgan) (after [48.181]). This walker has knees and arms and is perhaps the most human-like passive-dynamic walker to date (photo reproduced with permissions from [48.180])

Avidbots Ex1016
Page 1251 of 2154

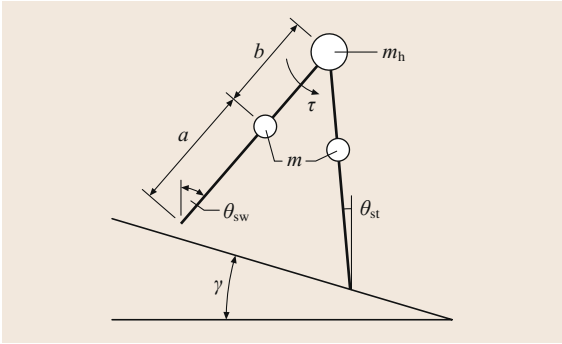


Fig. 48.10 The compass gait

lost at each heel strike, passive dynamic walkers require gravitational force to restore energy to the system before the next step. By choosing control inputs that regulate mechanical energy, for example by simulating gravity, walking on level terrain can be achieved [48.187, 188]. However, such approaches are still largely limited to modest terrains and periodic motions.

Trajectory optimization methods offer the potential to move beyond limit cycle gaits and to more complicated systems without passive stability properties. However, the fact that walking systems must make and break contact with the world complicates the trajectory optimization problem. Simulating rigid-body contact forces continuously using spring and damper models often results in stiff differential equations that can lead to numerical and computational issues. An alternative approach is to capture discontinuities in velocity by modeling collisions as impulsive events using a hybrid system representation (Sect. 48.2.3). For a walking hybrid system, a set of autonomous *guard functions* are defined such that $\phi_i(\boldsymbol{q}) = 0$ implies that body i is in contact and $\phi_i(\boldsymbol{q}) > 0$ implies the opposite. When a body comes into contact, the generalized velocity undergoes an instantaneous change, $\dot{\boldsymbol{q}}^+ = f_\Delta(\boldsymbol{q}, \dot{\boldsymbol{q}}^-)$, where $\dot{\boldsymbol{q}}^-$ and $\dot{\boldsymbol{q}}^+$ are the velocities pre and post-impact, respectively. The *mode* of the system is defined by the set of active contacts, $C = \{i | \phi_i(\boldsymbol{q}) = 0\}$, which in turn defines the set of Jacobian matrices in (48.2).

The hybrid system representation creates several challenges for trajectory optimization methods. The discontinuities due to mode transitions are typically accounted for explicitly using either *direct* or *multiple-shooting* methods [48.189]. Since each mode implies a different dynamics and mode changes are necessary for walking, this leads to the question of how to formulate the dynamics constraints. One solution is to predefine the sequence of mode transitions. For simple models with point feet, this approach can work quite well. However, for more complex models with many contacts, the number of modes grows exponentially, making mode sequence prespecification impractical.

The complementarity-based methods for simulating rigid body systems in contact described in Sect. 48.2.3 suggest another approach. By discretizing the dynamics in time and considering only the integral of contact forces over a time interval, the forward dynamics of a walking system can be greatly simplified. Since many trajectory optimization methods already discretize by evaluating a finite set of points along a trajectory, incorporation of contact forces as optimization variables leads to trajectory optimization as a nonlinear optimization problem with complementarity constraints [48.37].

Many impressive examples of trajectory optimization for walking and running exist in the literature. For example, *Mombaur* [48.78] simultaneously optimized motion and physical parameters for simulated bipeds to achieve open-loop stable running and *Remy* [48.190] applied trajectory optimization to generate efficient running for a quadruped. There are many more examples and it remains an active area of research. The problem of stabilizing trajectories optimized offline remains a challenge, particularly when the execution deviates from the planned mode sequence. Computational requirements still prevent nonlinear trajectory optimization methods from being applied online as an MPC algorithm, but improvements in numerical solvers and computing hardware will likely continue to reduce this gap.

48.6.2 Orbital Trajectory Stabilization and Control

Once an open-loop gait is generated, it must typically be stabilized in order to be executed on a robot. Classical techniques for trajectory stabilization from linear control can work, but a strong theme in stabilization for legged robots has been the idea that one should not enforce the precise timing of the trajectory. Section 48.3.2 discussed the notion of orbital stability and introduced Poincaré maps as a tool for stability analysis. Orbital stabilization, rather than traditional trajectory stabilization, sacrifices little and appears to be more compatible with underactuated robots. These ideas have also played an important role in developing stable dynamic walking and running controllers using *virtual constraints* [48.38]. Virtual constraints are holonomic constraints on monotonic functions of the robot's configuration variables. For example, for a forward periodic gait, it might be reasonable to assume that the angle of the stance leg with respect to the ground θ_{st} is monotonically increasing in time throughout the stance phase. The remaining configuration variables are written as functions of θ_{st}, effectively reparametrizing time, and constraints are imposed that, e.g., enforce sym-

metry in stance and swing leg angles ($\theta_{sw} = -\theta_{st}$) or coordinate arm and leg motion. This leads to outputs

$$y = \begin{bmatrix} h_1(q) \\ \cdots \\ h_d(q) \end{bmatrix}, \tag{48.72}$$

where each $h_i(q)$ encodes a virtual constraint. For example, $h(q) = \theta_{sw} + \theta_{st}$ encodes the virtual constraint for swing-stance leg symmetry. By driving the output dynamics asymptotically to 0 using high-gain control, the system can be seen to evolve according to the virtual constraints, yielding a lower dimensional problem for control design and stability analysis. This approach has been used to produce remarkable dynamic walking and running examples in real robots [48.191].

Poincaré maps are defined with respect to a transverse surface at a single point on a periodic orbit, which limits their usefulness for designing stabilizing controllers. The *transverse dynamics*, on the other hand, is defined with respect to a *continuous* family of surfaces $S(t)$, transverse to the orbit [48.45]. In the neighborhood of these coordinates, the dynamics can be written as

$$\dot{\tau} = 1 + f_1(\tau, x_\perp) \tag{48.73}$$

$$\dot{x}_\perp = A(\tau)x_\perp + f_2(\tau, x_\perp)\,, \tag{48.74}$$

where $\tau \in [0, t_f]$ represents the orbit phase, $x_\perp$ are the $2N-1$ coordinates orthogonal to the flow of the system, and $f_1(\cdot), f_2(\cdot)$ are functions containing nonlinear terms. Note that unlike the discrete Poincaré map, these dynamics do not require the trajectory of interest to be a periodic orbit.

By extracting the linear part of the transversal dynamics and incorporating control inputs, we obtain the transverse linearization

$$\dot{z}(t) = A(t)z(t) + B(t)u(t)\,, \tag{48.75}$$

for $t \in [0, t_f]$. Stabilization of this system, and therefore the underlying trajectory, can be approached using standard techniques from linear control theory. For walking systems with impulses, the dynamics can be extended to include a linearized impact map, $z(t_i^+) = \mathbf{C}_i z(t_i^-)$, where $t_i, i \in C$ are times at which impacts occur. Figure 48.11 illustrates the idea of transversal surfaces about a trajectory for a system with impacts. Manchester et al. [48.50] showed that under some mild assumptions, local exponential orbital stability of planned motions on a hybrid nonlinear system can be achieved using a receding horizon control with the transverse linear dynamics.

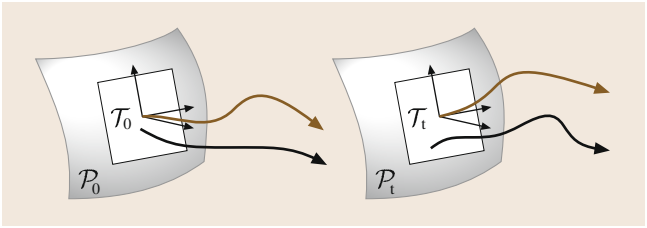


Fig. 48.11 A simple illustration of transverse linearization along a nominal trajectory (*red*) for systems with impacts

Minor variations in impact timing can be handled in linearization-based approaches to stabilizing walking trajectories, but perturbations that lead to changes in mode ordering often cannot. To address this problem, we need algorithms for adjusting walking trajectories online. As described in the previous section, the hybrid nature of walking systems makes this challenging. One way to avoid this issue is to employ smooth approximations of the dynamics that permit the use of efficient local trajectory optimization techniques at the expense of violating complementarity condition (48.22) by, e.g., allowing interpenetration of rigid bodies. This approach has led to impressive examples of online trajectory optimization for simulated systems [48.192], but it is yet unknown how well these approximations will transfer to physical systems.

48.7 Different Contact Behaviors

Contact forces have appeared throughout this chapter to be central for the modeling and control of legged robots. As a result, a different contact behavior can result in a significantly different dynamical behavior of the whole robot, and this can be extremely useful in situations where standard walking or running is inefficient or downright impossible.

48.7.1 Wall Climbing

Wall climbing legged robots rely on various devices such as suction cups, magnets, adhesive materials, miniature spine arrays, that can generate adhesive forces in order to stick to various vertical surfaces such as glass, steel, concrete, brick, stone [48.193–195]. In this case, the unilaterality condition (48.11), which was central throughout this chapter, is no more relevant. As a result, the whole locomotion modeling and control is deeply transformed.

48.7.2 Tethered Walking

On steep slopes, rappelling is an interesting option, tethering the robot to anchors in order to avoid tumbling

down. This has been experienced successfully on construction sites [48.196], or in a volcano in the case of the impressive Carnegie Mellon University (CMU) Dante II octopod robot [48.197]. In this case, the locomotion problem is similar to the general case described throughout this chapter, but the tether introduces an additional contact between the robot and the environment, on which the robot can pull but not push, so a unilateral contact exactly opposite to the standard unilateral contact between the feet and the ground, what can be treated however in exactly the same way.

48.7.3 Legs with Wheels

Adapting wheeled vehicles to rough terrain has led in some cases to implant wheels on legs, combining the flexibility of articulated legs on rough terrain with the efficiency of wheels on flat terrain. Different options are possible, passive or active wheels, passive or active legs [48.198–202]. In this case, the unilaterality condition (48.11) is fundamental like in the rest of this chapter, but the contact points can move freely on the contact surfaces along the direction of the wheels, what can significantly increase the array of possible motions [48.203]. Problems similar to those of standard legged robots include finding stable postures and allocating contact forces for maximizing stability and efficiency [48.200–202]. Interestingly, the case of passive wheels results in motions similar to skating [48.198].

48.7.4 Wheels with Legs

Evolving the mechanical design of legs for greater simplicity and improved efficiency has led in some cases to directly implant the legs on wheels [48.204, 205]. This results in very impressive outdoor performance, but not much related to standard legged robots in the end.

48.8 Conclusion

Legged systems are at the heart of some of the most exciting work in modern robotics. They offer the opportunity to travel to places beyond the reach of wheeled systems and gain fundamental insights into the conditions under which stable and efficient locomotion is possible. At the same time, their complex dynamics pose significant challenges for our computational approaches to control and stability analysis. Indeed, the simple fact that every walking system must engage in intermittent contact with its environment has significant mathematical implications.

A large number of success stories for legged systems moving with remarkable precision and reliability have been witnessed over the past decade, and the development of exciting new tools for optimization-based planning, control, and analysis offers the promise to develop machines that achieve even higher levels of robustness and efficiency.

References

48.1 K.J. Waldron, R.B. McGhee: The adaptive suspension vehicle, IEEE Control Syst. Mag. **6**, 7–12 (1986)

48.2 K. Hirai, M. Hirose, Y. Haikawa, T. Takenaka: The development of honda humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 1321–1326

48.3 H. Lim, A. Takanishi: Biped walking robots created at waseda university: WL and WABIAN family, Philos. Trans. Royal Soc. A **365**(1850), 49–64 (2007)

48.4 H. Miura, I. Shimoyama: Dynamic walk of a biped, Int. J. Robotics Res. **3**(2), 60–74 (1984)

48.5 M. Raibert: *Legged Robots that Balance* (MIT Press, Cambridge 1986)

48.6 T. McGeer: *Passive Dynamic Walking*, Simon Fraser University Tech. Rep., (Simon Fraser Univ., Burnaby 1988)

48.7 M. Raibert, K. Blankespoor, G. Nelson, R. Playter, the BigDog Team: BigDog, the rough-terrain quadruped robot, Proc. 17th World Cong. Int. Fed. Autom. Control. (2008)

48.8 P.-B. Wieber: Holonomy and nonholonomy in the dynamics of articulated motion, Proc. Ruperto Carola Symp. Fast Motion Biomech. Robotics (2005)

48.9 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)

48.10 M.K. Vukobratović: Contribution to the study of anthropomorphic systems, Kybernetika **8**(5), 404–418 (1972)

48.11 P. Sardain, G. Bessonnet: Forces acting on a biped robot. center of pressure–zero moment point, IEEE Trans. Syst. Man. Cybern. A **34**(5), 630–637 (2004)

48.12 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: ZMP analysis for arm/leg coordination, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 75–81

48.13 K. Harada, H. Hirukawa, F. Kanehiro, K. Fujiwara, K. Kaneko, S. Kajita, M. Nakamura: Dynamical

balance of a humanoid robot grasping an environment, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 1167–1173
48.14 Y. Or, E. Rimon: Analytic characterization of a class of 3-contact frictional equilibrium postures in 3D gravitational environments, Int. J. Robotics Res. **29**(1), 3–22 (2010)
48.15 P.-B. Wieber: On the stability of walking systems, Proc. Int. Workshop Humanoids Hum. Friendly Robots (2002)
48.16 T. Saida, Y. Yokokoji, T. Yoshikawa: FSW (feasible solution of wrench) for multi-legged robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)
48.17 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots – adios ZMP, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1976–1983
48.18 T. Bretl, S. Lall: Testing static equilibrium for legged robots, IEEE Trans. Robotics **24**(4), 794–807 (2008)
48.19 S. Barthélemy, P. Bidaud: Stability measure of postural dynamic equilibrium based on residual radius, Proc. Int. Symp. Adv. Robot Kinemat. (2008)
48.20 Z. Qiu, A. Escande, A. Micaelli, T. Robert: Human motions analysis and simulation based on a general criterion of stability, Proc. Int. Symp. Digit. Hum. Model. (2011)
48.21 Z. Qiu, A. Escande, A. Micaelli, T. Robert: A hierarchical framework for realizing dynamically-stable motions of humanoid robot in obstacle-cluttered environments, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)
48.22 E. Rimon, R. Mason, J.W. Burdick, Y. Or: A general stance stability test based on stratified morse theory with application to quasi-static locomotion planning, IEEE Trans. Robotics **24**(3), 626–641 (2008)
48.23 Q. Huang, S. Sugano, K. Tanie: Stability compensation of a mobile manipulator by manipulator motion: Feasibility and planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 1285–1292
48.24 J. Kim, W.K. Chung, Y. Youm, B.H. Lee: Real-time ZMP compensation method using null motion for mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1967–1972
48.25 B. Brogliato: *Nonsmooth Mechanics*, Communications and Control Engineering (Springer, London 1999)
48.26 F. Pfeiffer, C. Glocker: *Multibody Dynamics with Unilateral Contacts* (Wiley, New York 1996)
48.27 S. Chareyron, P.-B. Wieber: *Stability and Regulation of Nonsmooth Dynamical Systems* INRIA Res. Rep. RR-5408 (INRIA, Montbonnot Saint-Ismier 2004)
48.28 C. Liu, Z. Zhao, B. Brogliato: Frictionless multiple impacts in multibody systems. I. Theoretical framework, Proc. R. Soc. A **464**, 3193–3211 (2008)
48.29 Y.-B. Jia, M. Mason, M. Erdmann: Multiple impacts: A state transition diagram approach, Int. J. Robotics Res. **32**(1), 84–114 (2013)
48.30 J.-M. Bourgeot, C. Canudas de Wit, B. Brogliato: Impact shaping for double support walk: From the rocking block to the biped robot, Proc. Int. Conf. Climb. Walk. Robots (2005)
48.31 B. Gamus, Y. Or: Analysis of dynamic bipedal robot locomotion with stick-slip transitions, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 3348–3355
48.32 S. Kajita, K. Miura, M. Morisawa, K. Kaneko, F. Kanehiro, K. Yokoi: Evaluation of a stabilizer for biped walk with toe support phase, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)
48.33 V. Acary, B. Brogliato: *Numerical Methods for Nonsmooth Dynamical Systems*, Lect. Notes Appl. Comput. Mech., Vol. 35 (Springer, Berlin, Heidelberg 2008)
48.34 R.I. Leine, N. van de Wouw: *Stability and Convergence of Mechanical Systems with Unilateral Constraints*, Lect. Notes Appl. Comput. Mech., Vol. 36 (Springer, Berlin, Heidelberg 2008)
48.35 Y. Or, A.D. Ames: Stability and completion of zeno equilibria in lagrangian hybrid systems, IEEE Trans. Autom. Control **56**(6), 1322–1336 (2011)
48.36 D.E. Stewart, J.C. Trinkle: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction, Int. J. Numer. Methods Eng. **39**(15), 2673–2691 (1996)
48.37 M. Posa, C. Cantu, R. Tedrake: A direct method for trajectory optimization of rigid bodies through contact, Int. J. Robotics Res. **33**(1), 69–81 (2014)
48.38 E.R. Westervelt, J.W. Grizzle, C. Chevallereau, J.H. Choi, B. Morris: *Feedback Control of Dynamic Bipedal Robot Locomotion* (CRC, Boca Raton 2007)
48.39 M. Wisse: Essentials of Dynamic Walking: Analysis and Design of Two-Legged Robots, Dissertation (Technische Universiteit, Delft 2004)
48.40 R. Tedrake, I.R. Manchester, M.M. Tobenkin, J.W. Roberts: LQR-Trees: Feedback motion planning via sums of squares verification, Int. J. Robotics Res. **29**, 1038–1052 (2010)
48.41 M. Posa, M. Tobenkin, R. Tedrake: Lyapunov analysis of rigid body systems with impacts and friction via sums-of-squares, Proc. Int. Conf. Hybrid Syst. Comput. Control (2013) pp. 63–72
48.42 S. Cotton, I. Olaru, M. Bellman, T. van der Ven, J. Godowski, J. Pratt: Fastrunner: A fast, efficient and robust bipedal robot. concept and planar simulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)
48.43 M. Srinivasan, A. Ruina: Computer optimization of a minimal biped model discovers walking and running, Nature **439**, 72–75 (2006)
48.44 K. Mombaur, H.G. Bock, J.P. Schloder, R.W. Longman: Open-loop stable solutions of periodic optimal control problems in robotics, Z. Angew. Math. Mech. **85**(7), 499–515 (2005)
48.45 J. Hauser, C.C. Chung: Converse Lyapunov functions for exponentially stable periodic orbits, Syst. Control Lett. **23**(1), 27–34 (1994)

Avidbots Ex1016
Page 1255 of 2154

48.46 J. Guckenheimer, P. Holmes: *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields* (Springer, Berlin, Heidelberg 1983)

48.47 I.R. Manchester, M.M. Tobenkin, M. Levashov, R. Tedrake: Regions of attraction for hybrid limit cycles of walking robots, Proc. 21st World Cong. Int. Fed. Autom. Control (2011)

48.48 E.R. Westervelt, G. Buche, J.W. Grizzle: Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds, Int. J. Robotics Res. **23**(6), 559–582 (2004)

48.49 A.S. Shiriaev, L.B. Freidovich, I.R. Manchester: Can we make a robot ballerina perform a pirouette? Orbital stabilization of periodic motions of underactuated mechanical systems, Annu. Rev. Control **32**(2), 200–211 (2008)

48.50 I.R. Manchester, U. Mettin, F. Iida, R. Tedrake: Stable dynamic walking over uneven terrain, Int. J. Robotics Res. **30**(3), 265–279 (2011)

48.51 J.-P. Aubin: *Viability Theory* (Birkhäuser, Basel 1991)

48.52 P.-B. Wieber: Constrained dynamics and parametrized control in biped walking, Proc. Int. Symp. Math. Theory Networks Syst. (2000)

48.53 P.-B. Wieber: Viability and predictive control for safe locomotion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008)

48.54 K. Ogata: *Modern Control Engineering*, 3rd edn. (Prentice Hall, Upper Saddle River 1996)

48.55 J. Pratt, R. Tedrake: Velocity based stability margins for fast bipedal walking, Proc. Ruperto Carola Symp. Fast Motion Biomech. Robotics (2005)

48.56 A.L. Hof, M.G.J. Gazendam, W.E. Sinke: The condition for dynamic stability, J. Biomech. **38**, 1–8 (2005)

48.57 J. Pratt, J. Carff, S. Drakunov, A. Goswami: Capture point: A step toward humanoid push recovery, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.58 T. Takenaka, T. Matsumoto, T. Yoshiike: Real time motion generation and control for biped robot – 1st report: Walking gait pattern generation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009)

48.59 J. Englsberger, C. Ott, M.A. Roa, A. Albu-Schäffer, G. Hirzinger: Bipedal walking control based on capture point dynamics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011)

48.60 T. Koolen, T. de Boer, J. Rebula, A. Goswami, J. Pratt: Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models, Int. J. Robotics Res. **31**(9), 1094–1113 (2012)

48.61 A. Papachristodoulou, S. Prajna: Robust stability analysis of nonlinear hybrid systems, IEEE Trans. Autom. Control **54**(5), 1035–1041 (2009)

48.62 P. Hanggi, P. Talkner, M. Borkovec: Reaction-rate theory: Fifty years after Kramers, Rev. Mod. Phys. **62**(2), 251–342 (1990)

48.63 K. Byl: Metastable Legged-Robot Locomotion, Dissertation (MIT, Cambridge 2008)

48.64 J. Steinhardt, R. Tedrake: Finite-time regional verification of stochastic nonlinear systems, Int. J. Robotics Res. **31**(7), 901–923 (2012)

48.65 D.G.E. Hobbelen, M. Wisse: A disturbance rejection measure for limit cycle walkers: The gait sensitivity norm, IEEE Trans. Robotics **23**(6), 1213–1224 (2007)

48.66 C. Ebenbauer: Polynomial Control Systems: Analysis and Design via Dissipation Inequalities and Sum of Squares, Dissertation (Univ. Stuttgart, Stuttgart 2005)

48.67 H. Dai, R. Tedrake: L2-gain optimization for robust bipedal walking on unknown terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

48.68 E. Garcia, J. Estremera, P. Gonzalez de Santos: A classification of stability margins for walking robots, Proc. Int. Conf. Climb. Walk. Robots (2002)

48.69 A. Goswami: Postural stability of biped robots and the foot rotation indicator (FRI) point, Int. J. Robotics Res. **18**(6), 523–533 (1999)

48.70 C.K. Chow, D.H. Jacobson: *Studies of human locomotion via optimal programming*, Tech. Rep. No. 617 (Harvard Univ., Cambridge 1970)

48.71 P.H. Channon, S.H. Hopkins, D.T. Phan: Derivation of optimal walking motions for a biped walking robot, Robotica **10**(2), 165–172 (1992)

48.72 G. Cabodevilla, N. Chaillet, G. Abba: Energy-minimized gait for a biped robot, Proc. Auton. Mob. Syst. (1995)

48.73 C. Chevallereau, A. Formal'sky, B. Perrin: Low energy cost reference trajectories for a biped robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

48.74 M. Rostami, G. Bessonnet: Impactless sagittal gait of a biped robot during the single support phase, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

48.75 L. Roussel, C. Canudas de Wit, A. Goswami: Generation of energy optimal complete gait cycles for biped robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

48.76 J. Denk, G. Schmidt: Synthesis of a walking primitive database for a humanoid robot using optimal control techniques, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2001)

48.77 T. Buschmann, S. Lohmeier, H. Ulbrich, F. Pfeiffer: Optimization based gait pattern generation for a biped robot, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2005)

48.78 K. Mombaur: Using optimization to create self-stable human-like running, Robotica **27**(3), 321–330 (2009)

48.79 J. Denk, G. Schmidt: Synthesis of walking primitive databases for biped robots in 3D-environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

48.80 S.A. Setiawan, S.H. Hyon, J. Yamaguchi, A. Takanishi: Quasi real-time walking control of a bipedal humanoid robot based on walking pattern synthesis, Proc. Int. Symp. Exp. Robotics (1999)

48.81 P.-B. Wieber, C. Chevallereau: Online adaptation of reference trajectories for the control of walking systems, Robotics Auton. Syst. **54**(7), 559–566 (2006)

48.82 C. Liu, C.G. Atkeson: Standing balance control using a trajectory library, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3031–3036

Avidbots Ex1016
Page 1256 of 2154

48.83 J. Yamaguchi, A. Takanishi, I. Kato: Development of biped walking robot compensating for three-axis moment by trunk motion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1993)

48.84 Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, K. Tanie: Planning walking patterns for a biped robot, IEEE Trans. Robotics Autom. **17**(3), 280–289 (2001)

48.85 P.-B. Wieber: Trajectory free linear model predictive control for stable walking in the presence of strong perturbations, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.86 T. de Boer: Foot placement in robotic bipedal locomotion, Dissertation (Technische Univ. Delft, Dleft 2012)

48.87 R.J. Full, D.E. Koditschek: Templates and anchors: Neuromechanical hypotheses of legged locomotion on land, J. Exp. Biol. **202**, 3325–3332 (1999)

48.88 R.M. Alexander: Mechanics of bipedal locomotion, Persp. Exp. Biol. **1**, 493–504 (1976)

48.89 R.M. Alexander: Simple models of human movement, ASME Appl. Mech. Rev. **48**(8), 461–470 (1995)

48.90 S.S. Keerthi, E.G. Gilbert: Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations, J. Optim. Theory Appl. **57**(2), 265–293 (1988)

48.91 D.Q. Mayne, J.B. Rawlings, C.V. Rao, P.O.M. Scokaert: Constrained model predictive control: Stability and optimality, Automatica **26**(6), 789–814 (2000)

48.92 M. Alamir, N. Marchand: Numerical stabilisation of non-linear systems: Exact theory and approximate numerical implementation, Eur. J. Control **5**(1), 87–97 (1999)

48.93 M. Alamir, G. Bornard: Stability of a truncated infinite constrained receding horizon scheme: The general discrete nonlinear case, Automatica **31**(9), 1353–1356 (1995)

48.94 A. Takanishi, M. Tochizawa, H. Karaki, I. Kato: Dynamic biped walking stabilized with optimal trunk and waist motion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1989)

48.95 H. Lim, Y. Kaneshima, A. Takanishi: Online walking pattern generation for biped humanoid robot with trunk, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

48.96 T. Buschmann, S. Lohmeier, M. Bachmayer, H. Ulbrich, F. Pfeiffer: A collocation method for real-time walking pattern generation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2007)

48.97 K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, H. Inoue: Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired ZMP, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002)

48.98 R. Tajima, D. Honda, K. Suga: Fast running experiments involving a humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009)

48.99 K. Nagasaka, Y. Kuroki, S. Suzuki, Y. Itoh, J. Yamaguchi: Integrated motion control for walking, jumping and running on a small bipedal entertainment robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

48.100 M. Morisawa, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, S. Nakaoka, H. Hirukawa: A biped pattern generation allowing immediate modification of foot placement in real-time, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.101 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Biped walking pattern generation by using preview control of zero moment point, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1620–1626

48.102 K. Nishiwaki, S. Kagami: Online walking control systems for humanoids with short cycle pattern generation, Int. J. Robotics Res. **28**(6), 729–742 (2009)

48.103 J. Park, Y. Youm: General ZMP preview control for bipedal walking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)

48.104 D. Gouaillier, C. Collette, C. Kilner: Omni-directional closed-loop walk for NAO, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2010)

48.105 M. Krause, J. Englsberger, P.-B. Wieber, C. Ott: Stabilization of the capture point dynamics for bipedal walking based on model predictive control, Proc. IFAC Symp. Robot Control (2012)

48.106 M. van de Panne: From footprints to animation, Comput. Graph. **16**(4), 211–223 (1997)

48.107 A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, M. Diehl: Online walking motion generation with automatic foot step placement, Adv. Robotics **24**(5-6), 719–737 (2010)

48.108 A. Herdt, N. Perrin, P.-B. Wieber: LMPC based online generation of more efficient walking motions, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)

48.109 Z. Aftab, T. Robert, P.-B. Wieber: Ankle, hip and stepping strategies for humanoid balance recovery with a single model predictive control scheme, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)

48.110 J. Urata, K. Nishiwaki, Y. Nakanishi, K. Okada, S. Kagami, M. Inaba: Online decision of foot placement using singular LQ preview regulation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2011)

48.111 J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, H. Inoue: Dynamically-stable motion planning for humanoid robots, Auton. Robots **12**, 105–118 (2002)

48.112 S. Dalibard, A. El Khoury, F. Lamiraux, M. Taix, J.-P. Laumond: Small-space controllability of a walking humanoid robot, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2011)

48.113 K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, T. Isozumi: Humanoid robot HRP-2, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 1083–1090

48.114 E. Yoshida, C. Esteves, I. Belousov, J.-P. Laumond, T. Sakaguchi, K. Yokoi: Planning 3-D collision-free dynamic robotic motion through iterative reshaping, IEEE Trans. Robotics **24**(5), 1186–1198 (2008)

Avidbots Ex1016
Page 1257 of 2154

48.115 J.-D. Boissonnat, O. Devillers, S. Lazard: Motion planning of legged robots, SIAM J. Comput. **30**(1), 218–246 (2000)

48.116 N. Perrin, O. Stasse, F. Lamiraux, E. Yoshida: Weakly collision-free paths for continuous humanoid footstep planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011)

48.117 N. Perrin: From discrete to continuous motion planning, Proc. Int. Workshop Algorithm. Found. Robotics (2012)

48.118 J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, H. Inoue: Online footstep planning for humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

48.119 J. Chestnutt, J. Kuffner: A tiered planning strategy for biped navigation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2004)

48.120 P. Michel, J. Chestnutt, J. Kuffner, T. Kanade: Vision-Guided Humanoid Footstep Planning for Dynamic Environments. In: *Proc. IEEE/RAS Int. Conf. Humanoid Robots* 2005)

48.121 J. Chestnutt, P. Michel, J. Kuffner, T. Kanade: Locomotion among dynamic obstacles for the honda asimo, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)

48.122 J.-M. Bourgeot, N. Cislo, B. Espiau: Path-planning and tracking in a 3D complex environment for an anthropomorphic biped robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002)

48.123 J. Chestnutt, J. Kuffner, K. Nishiwaki, S. Kagami: Planning biped navigation strategies in complex environments, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2003)

48.124 M. Zucker, J.A. Bagnell, C. Atkeson, J. Kuffner: An optimization approach to rough terrain locomotion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

48.125 J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, S. Kagami: Biped navigation in rough environments using on-board sensing, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009)

48.126 N. Perrin, O. Stasse, F. Lamiraux, Y. Kim, D. Manocha: Real-time footstep planning for humanoid robots among 3D obstacles using a hybrid bounding box, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

48.127 N. Perrin, O. Stasse, L. Baudoin, F. Lamiraux, E. Yoshida: Fast humanoid robot collision-free footstep planning using swept volume approximations, IEEE Trans. Robotics **28**(2), 427–439 (2012)

48.128 R.L.H. Deits, R. Tedrake: Computing large convex regions of obstacle-free space through semidefinite programming, Proc. Int. Workshop Algorithmic Found. Robotics (2014)

48.129 T. Bretl, S. Lall, J.-C. Latombe, S. Rock: Multi-step motion planning for free-climbing robots, Proc. Int. Workshop Algorithmic Found. Robotics (2004)

48.130 K. Hauser, T. Bretl, J.-C. Latombe: Non-gaited humanoid locomotion planning, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2005)

48.131 K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, B. Wilcox: Motion planning for legged robots on varied terrain, Int. J. Robotics Res. **27**(11/12), 1325–1349 (2008)

48.132 A. Escande, A. Kheddar, S. Miossec: Planning support contact-points for humanoid robots and experiments on HRP-2, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

48.133 A. Escande, A. Kheddar, S. Miossec, S. Garsault: Planning support contact-points for acyclic motions and experiments on HRP-2, Proc. Int. Symp. Exp. Robotics (2008)

48.134 K. Bouyarmane, J. Vaillant, F. Keith, A. Kheddar: Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)

48.135 S. Lengagne, J. Vaillant, E. Yoshida, A. Kheddar: Generation of whole-body optimal dynamic multi-contact motions, Int. J. Robotics Res. **32**(9/10), 1104–1119 (2013)

48.136 K. Harada, E. Yoshida, K. Yokoi: *Motion Planning for Humanoid Robots* (Springer, Berlin, Heidelberg 2010)

48.137 R.B. McGhee, A.A. Frank: On the stability properties of quadruped creeping gaits, Math. Biosci. **3**, 331–351 (1968)

48.138 G.C. Haynes, A.A. Rizzi: Gaits and gait transitions for legged robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1117–1122

48.139 Y. Fujimoto, A. Kawamura: Proposal of biped walking control based on robust hybrid position/force control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneap. (1996) pp. 2724–2730

48.140 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, H. Hirukawa: A realtime pattern generator for biped walking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 31–37

48.141 K. Yin, K. Loken, M. van de Panne: SIMBICON: Simple Biped Locomotion Control, Proc. ACM SIGGRAPH (2007)

48.142 I.-W. Park, J.-Y. Kim, J.-H. Oh: Online biped walking pattern generation for humanoid robot KHR-3(KAIST humanoid robot – 3: HUBO), Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.143 A. Ijspeert, A. Crespi, D. Ryczko, J.-M. Cabelguen: From swimming to walking with a salamander robot driven by a spinal cord model, Science **315**(5817), 1416–1420 (2007)

48.144 R. Katoh, M. Mori: Control method of biped locomotion giving asymptotic stability of trajectory, Automatica **20**(4), 405–414 (1984)

48.145 L. Righetti, A. Ijspeert: Programmable central pattern generators: An application to biped locomotion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

48.146 K. Matsuoka: Sustained oscillations generated by mutually inhibiting neurons with adaptation, Biol. Cybern. **52**, 367–376 (1985)

48.147 G. Endo, J. Morimoto, J. Nakanishi, G. Cheng: An empirical exploration of a neural oscillator for

Avidbots Ex1016
Page 1258 of 2154

biped locomotion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 3036–3042
48.148 Y. Fukuoka, H. Kimura, A. Cohen: Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts, Int. J. Robotics Res. **22**(3–4), 187–202 (2003)
48.149 R. Brooks: Elephants don't play chess, Robotics Auton. Syst. **6**, 3–15 (1990)
48.150 R. Brooks: A robot that walks; emergent behaviors from a carefully evolved network, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 292–296
48.151 J. Pratt, C.-M. Chew, A. Torres, P. Dilworth, G. Pratt: Virtual model control: An intuitive approach for bipedal locomotion, Int. J. Robotics Res. **20**, 129–143 (2001)
48.152 F. Génot, B. Espiau: On the control of the mass center of legged robots under unilateral constraints, Proc. Int. Conf. Climb. Walk. Robots (1998)
48.153 L. Saab, O.E. Ramos, F. Keith, N. Mansard, P. Souères, J.-Y. Fourquet: Dynamic whole-body motion generation under rigid contacts and other unilateral constraints, IEEE Trans. Robotics **29**(2), 346–362 (2013)
48.154 L. Sentis, J. Park, O. Khatib: Compliant control of multicontact and center-of-mass behaviors in humanoid robots, IEEE Trans. Robotics **26**(3), 483–501 (2010)
48.155 R.W. Brockett: Asymptotic stability and feedback stabilization. In: *Differential Geometric Control Theory*, (Birkhäuser, Boston 1983)
48.156 S. Kajita, T. Nagasaki, K. Kaneko, K. Yokoi, K. Tanie: A running controller of humanoid biped HRP-2LR, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 618–624
48.157 L. Sentis, O. Khatib: Control of free-floating humanoid robots through task prioritization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1730–1735
48.158 A. De Luca, G. Oriolo: Modelling and control of nonholonomic mechanical systems, CISM Int. Centre Mech. Sci. **360**, 277–342 (1995)
48.159 Y. Nakamura, R. Mukherjee: Exploiting nonholonomic redundancy of free-flying space robots, IEEE Trans. Robotics Autom. **9**(4), 499–506 (1993)
48.160 E. Papadopoulos: Nonholonomic behaviour in free-floating space manipulators and its utilization. In: *Nonholonomic Motion Planning*, ed. by Y. Xu, T. Kanade (Kluwer Academic, Dordrecht 1993)
48.161 C. Chevallereau, E.R. Westervelt, J.W. Grizzle: Asymptotically stable running for a five-link, four-actuator, planar bipedal robot, Int. J. Robotics Res. **24**(6), 431–464 (2005)
48.162 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Resolved momentum control: Humanoid motion planning based on the linear and angular momentum, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1644–1650
48.163 M. Popovic, A. Hofmann, H. Herr: Zero spin angular momentum control: definition and applicability, IEEE/RAS Int. Conf. Humanoid Robots (2004)
48.164 S.-H. Lee, A. Goswami: Ground reaction force control at each foot: A momentum-based humanoid balance controller for non-level and non-stationary ground, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3157–3162
48.165 C. Ott, M.A. Roa, G. Hirzinger: Posture and balance control for biped robots based on contact force optimization, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2011)
48.166 T. Sugihara: Standing stabilizability and stepping maneuver in planar bipedalism based on the best COM-ZMP regulator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009)
48.167 Y. Choi, D. Kim, B.-J. You: On the walking control for humanoid robot based on the kinematic resolution of CoM jacobian with embedded motion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2655–2660
48.168 S.-H. Hyon: Compliant terrain adaptation for biped humanoids without measuring ground surface and contact forces, IEEE Trans. Robotics **25**(1), 171–178 (2009)
48.169 S. Kajita, K. Yokoi, M. Saigo, K. Tanie: Balancing a humanoid robot using backdrive concerned torque control and direct angular momentum feedback, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3376–3382
48.170 S. Lohmeier, K. Löffler, M. Gienger, H. Ulbrich, F. Pfeiffer: Computer system and control of biped *Johnnie*, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4222–4227
48.171 J.-H. Kim, J.-H. Oh: Walking control of the humanoid platform KHR-1 based on torque feedback control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 623–628
48.172 M.-S. Kim, J.-H. Oh: Posture control of a humanoid robot with a compliant ankle joint, Int. J. Humanoid Robotics **7**(1), 5–29 (2010)
48.173 T. Buschmann, S. Lohmeier, H. Ulbrich: Biped walking control based on hybrid position/force control, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3019–3024
48.174 S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, K. Yokoi: Biped walking stabilization based on linear inverted pendulum tracking, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 4489–4496
48.175 M. Morisawa, S. Kajita, F. Kanehiro, K. Kaneko, K. Miura, K. Yokoi: Balance control based on capture point error compensation for biped walking on uneven terrain, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)
48.176 Y. Fujimoto, A. Kawamura: Simulation of an autonomous biped walking robot including environmental force interaction, IEEE Robotics Autom. Mag. **5**(2), 33–41 (1998)
48.177 L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, S. Schaal: Optimal distribution of contact forces

Avidbots Ex1016
Page 1259 of 2154

with inverse dynamics control, Int. J. Robotics Res. **32**(3), 280–298 (2013)
48.178 A.D. Ames: Human-inspired control of bipedal robotics via control lyapunov functions and quadratic programs, Proc. Int. Conf. Hybrid Syst. Comput. Control (2013)
48.179 S. Kuindersma, F. Permenter, R. Tedrake: An efficiently solvable quadratic program for stabilizing dynamic locomotion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)
48.180 S.H. Collins, A. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive-dynamic walkers, Science **307**, 1082–1085 (2005)
48.181 S.H. Collins, M. Wisse, A. Ruina: A three-dimensional passive-dynamic walking robot with two legs and knees, Int. J. Robotics Res. **20**(7), 607–615 (2001)
48.182 R. Tedrake, T.W. Zhang, H.S. Seung: Stochastic policy gradient reinforcement learning on a simple 3D biped, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2849–2854
48.183 J.E. Wilson: Walking Toy, Patent 0 (1936)
48.184 R. Tedrake, T.W. Zhang, M.F. Fong, H.S. Seung: Actuating a simple 3D passive dynamic walker, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4656–4661
48.185 M. Garcia, A. Chatterjee, A. Ruina: Efficiency, speed, and scaling of two-dimensional passive-dynamic walking, Dyn. Stab. Sytems **15**(2), 75–99 (2000)
48.186 A. Goswami, B. Thuilot, B. Espiau: *Compass-Like Biped Robot Part I : Stability and Bifurcation of Passive Gaits*, INRIA Res. Rep. No. 2996 (INRIA, Les Chesnay Cedex 1996)
48.187 M.W. Spong, G. Bhatia: Further results on control of the compass gait biped, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1933–1938
48.188 F. Asano, Z.-W. Luo, M. Yamakita: Biped gait generation and control based on a unified property of passive dynamic walking, IEEE Trans. Robotics **21**(4), 754–762 (2005)
48.189 J.T. Betts: Survey of numerical methods for trajectory optimization, J. Guid. Control, Dyn. **21**(2), 193–207 (1998)
48.190 C.D. Remy: Optimal Exploitation of Natural Dynamics in Legged Locomotion, Dissertation (ETH, Zurich 2011)
48.191 K. Sreenath, H.W. Park, I. Poulakakis, J.W. Grizzle: A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal walking on MABEL, Int. J. Robotics Res. **30**(9), 1170–1193 (2011)
48.192 Y. Tassa, T. Erez, E. Todorov: Synthesis and stabilization of complex behaviors through online trajectory optimization, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4906–4913
48.193 S. Hirose, A. Nagakubo, R. Toyama: Machine that can walk and climb on floors, walls and ceilings, Proc. Int. Conf. Adv. Robotics (1991) pp. 753–758
48.194 T. Yano, S. Numao, Y. Kitamura: Development of a self-contained wall climbing robot with scanning type suction cups, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 249–254
48.195 S. Kim, A. Asbeck, W. Provancher, M.R. Cutkosky: SpinybotII: Climbing hard walls with compliant microspines, Proc. Int. Conf. Adv. Robotics (2005) pp. 18–20
48.196 S. Hirose, K. Yoneda, H. Tsukagoshi: TITAN VII: quadruped walking and manipulating robot on a steep slope, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 494–500
48.197 J. Bares, D. Wettergreen: Dante II: technical description, results and lessons learned, Int. J. Robotics Res. **18**(7), 621–649 (1999)
48.198 G. Endo, S. Hirose: Study on roller-walker (system integration and basic experiments), IEEE Int. Conf. Robotics Autom. (1999) pp. 2032–2037
48.199 R. Siegwart, P. Lamon, T. Estier, M. Lauria, R. Piguet: Innovative design for wheeled locomotion in rough terrain, Robotics Auton. Syst. **40**, 151–162 (2002)
48.200 K. Iagnemma, A. Rzepniewski, S. Dubowsky: Control of robotic vehicles with actively articulated suspensions in rough terrain, Auton. Robots **14**, 5–16 (2003)
48.201 K. Iagnemma, S. Dubowsky: Traction control of wheeled robotic vehicles in rough terrain with application to planetary rovers, Int. J. Robotics Res. **23**(10–11), 1029–1040 (2004)
48.202 C. Grand, F. Ben Amar, F. Plumet, P. Bidaud: Stability and traction optimization of a wheel-legged robot, Int. J. Robotics Res. **23**(10–11), 1041–1058 (2004)
48.203 G. Besseron, C. Grand, F. Ben Amar, F. Plumet, P. Bidaud: Locomotion modes of an hybrid wheel-legged robot, Proc. Int. Conf. Climb. Walk. Robots (2004)
48.204 U. Saranli, M. Buehler, D.E. Koditschek: Rhex - a simple and highly mobile hexapod robot, Int. J. Robotics Res. **20**(7), 616–631 (2001)
48.205 T. Allen, R. Quinn, R. Bachmann, R. Ritzmann: Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1370–1375

Avidbots Ex1016
Page 1260 of 2154

Multimedia Contents

49. Modeling and Control of Wheeled Mobile Robots

Claude Samson, Pascal Morin, Roland Lenain

This chapter may be seen as a follow up to Chap. 24, devoted to the classification and modeling of basic wheeled mobile robot (WMR) structures, and a natural complement to Chap. 47, which surveys motion planning methods for WMRs. A typical output of these methods is a feasible (or admissible) reference state trajectory for a given mobile robot, and a question which then arises is how to make the physical mobile robot track this reference trajectory via the control of the actuators with which the vehicle is equipped. The object of the present chapter is to bring elements of the answer to this question based on simple and effective control strategies.

The chapter is organized as follows. Section 49.2 is devoted to the choice of control models and the determination of modeling equations associated with the path-following control problem. In Sect. 49.3, the path following and trajectory stabilization problems are addressed in the simplest case when no requirement is made on the robot orientation (i. e., position control). In Sect. 49.4 the same problems are revisited for the control of both position and orientation. The previously mentionned sections consider an ideal robot satisfying the rolling-without-sliding assumption. In Sect. 49.5, we relax this assumption in order to take into account nonideal wheel-ground contact. This is especially important for field-robotics applications and the proposed results are validated through full scale experiments on natural terrain. Finally, a few complementary issues on the feedback control of mobile robots are briefly discussed in the concluding Sect. 49.6, with a list of commented references for further reading on WMRs motion control.

49.1 **Background** 1236
49.1.1 Path Following 1237
49.1.2 Stabilization of Trajectories 1237
49.1.3 Stabilization of Fixed Postures 1237

49.2 **Control Models** 1238
49.2.1 Kinematics Versus Dynamics 1238
49.2.2 Modeling in an Absolute Frame 1238
49.2.3 Modeling in a Frénet Frame 1239

49.3 **Adaptation of Control Methods for Holonomic Systems** 1240
49.3.1 Stabilization of Trajectories for a Nonconstrained Point 1240
49.3.2 Path Following with no Orientation Control 1240

49.4 **Methods Specific to Nonholonomic Systems** 1241
49.4.1 Transformation of Kinematic Models into the Chained Form 1242
49.4.2 Tracking of a Reference Vehicle with the Same Kinematics 1242
49.4.3 Path Following with Orientation Control 1244
49.4.4 Asymptotic Stabilization of Fixed Postures 1246
49.4.5 Limitations Inherent to the Control of Nonholonomic Systems 1250
49.4.6 Practical Stabilization of Arbitrary Trajectories 1251

49.5 **Path Following in the Case of Nonideal Wheel-Ground Contact** 1255
49.5.1 Extended Control Models in the Presence of Sliding 1255
49.5.2 On-Line Estimation of Sliding Angles 1257
49.5.3 Feedback Laws for Path Following 1259

Avidbots Ex1016
Page 1261 of 2154

49.5.4 Path Following Illustrations in Low Grip Conditions 1259
49.6 **Complementary Issues and Bibliographical Guide** 1261
49.6.1 General Trailer Systems 1261
49.6.2 Sensor-Based Motion Control 1262
49.6.3 Sliding Effects and Other Dynamical Issues 1262
49.6.4 Bibliographical Guide 1262
Video-References 1263
References 1263

49.1 Background

Wheeled mobile robotics remains a very active research topic, as witnessed by the Darpa Grand Challenges [49.1]. The first Grand Challenge, in 2005, took place in a clear environment and was won by Stanley (Fig. 49.1a). The second Grand Challenge, in 2007, took place in urban environment and demonstrated autonomous driving capabilities, especially in complex environments with strong safety and reliability issues. This made possible the development of autonomous cars, such as google car (Fig. 49.1b). Whatever the sensors used for localization and detection, the use in every day life of cybercars, such as VipaLab (Fig. 49.1c), is becoming a reality. Numerous tests are under progress all around the world showing the benefits of autonomous transportation systems. On-road transportation probably constitutes the main and most visible application domain of emerging autonomous wheeled robots. Driving assistance for human activities in off-road conditions constitutes another application domain. Indeed, WMRs can also be used in hazardous environments, or in environments that are not reachable by humans. Planetary exploration (Chap. 55), with the example of Curiosity (Fig. 49.1d), is a popular example of such applications. Other developments, concerning people well-being and safety, are also investigated in the field of environment monitoring, surveillance, agriculture, civil protection or defence. Figure 49.1e,f illustrate these developments with the first one representing the Claas etrion robot (for autonomous farming, see also Chap. 56) and the second one representing the SUGV robot manufactured by iRobot. This robot, dedicated to civil protection, was the first one to enter the damaged Fukushima power plant. All these examples highlight the possible benefits and different applications of WMRs. The variety of robot designs (in term of scale, locomotion mode, or related actuation strategy see Chap. 24), is in line with the diversity of application needs. Whatever the system and application, however, autonomy relies on the design of efficient feedback laws that can ensure precise autonomous motion of the vehicle despite all possible modeling errors and perturbations. This chapter is dedicated to this issue and aims at providing the basics of feedback control design for nonholonomic WMRs. Implementation of these control laws supposes that one is able to measure the variables involved in the control loop (typically the position and orientation of the mobile robot with respect to either a fixed frame or a path that the vehicle should follow). Throughout this chapter we will assume that these measurements are available continuously in time and that they are not corrupted by noise. In a general manner, robustness considerations will not be discussed in detail, one reason being that, beyond imposed space limitations, a large part of the presented approaches are based on linear control theory. The feedback control laws then inherit the strong robustness properties associated with stable linear systems. Results can also be subsequently refined by using

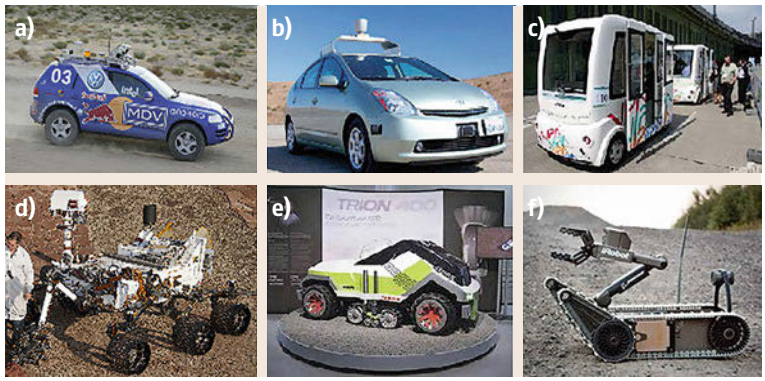


Fig.49.1a–f Some example of mobile robots for different applications: **(a)** Stanley (Stanford Univ.), off-road motion; **(b)** Google car, autonomous car; **(c)** Vipa (Institut Pascal/Ligier), urban transportation; **(d)** Curiosity (NASA) planetary exploration; **(e)** Etrion (Claas), agriculture robot; **(f)** PackBot (iRobot), public safety

Avidbots Ex1016
Page 1262 of 2154

complementary, eventually more elaborate, automatic control techniques. For the sake of simplicity, the control methods are developed mainly for unicycle-type and car-like mobile robots, which correspond respectively to the types $(2,0)$ and $(1,1)$ in the classification proposed in Chap. 24. Most of the results can in fact be easily extended/adapted to other mobile robots, in particular to systems with trailers. We will mention the cases where such extensions are straightforward. All reported simulation results, illustrating various control problems and solutions, are carried out for a car-like vehicle, whose kinematics is slightly more complex than that of unicycle-type vehicles.

Recall (Fig. 49.2) that:

1. A *unicycle-type mobile robot* is schematically composed of two independent actuated wheels on a common axle whose direction is rigidly linked to the robot chassis, and one or several passively orientable – or caster – wheels, which are not controlled and serve for sustentation purposes.
2. A (rear-drive) *car-like mobile robot* is composed of a motorized wheeled axle at the rear of the chassis, and one (or a pair of) orientable front steering wheel(s).

Note also, as illustrated by the diagram below (Fig. 49.3), that a car-like mobile robot can be viewed (at least kinematically) as a unicycle-type mobile robot to which a trailer is attached.

Despite the variety of application, three generic control problems can mainly be considered in mobile robotics and are detailed in this chapter.

49.1.1 Path Following

Given a curve C on the plane, a (nonzero) longitudinal velocity v_0 for the robot chassis, and a point P attached to the chassis, the goal is to have the point P follow the curve C when the robot moves with the velocity v_0. The variable that one has to stabilize at zero is thus the distance between the point P and the curve (i. e., the distance between P and the closest point M on C). This type of problem typically corresponds to driving on a road while trying to maintain the distance between the vehicle chassis and the side of the road constant. Automatic wall following is another possible application.

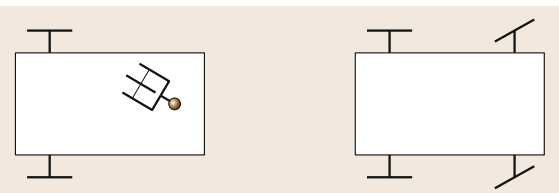


Fig.49.2a,b Unicycle-type (a) and car-like (b) mobile robots

49.1.2 Stabilization of Trajectories

This problem differs from the previous one in that the vehicle's longitudinal velocity is no longer predetermined because one also aims to monitor the distance gone along the curve C. This objective supposes that the geometric curve C is complemented with a time schedule, i. e., that it is parameterized with the time variable t. This boils down to defining a trajectory $t \longmapsto (x_r(t), y_r(t))$ with respect to a reference frame $\mathcal{F}_0$. Then the goal is to stabilize the position error vector $(x(t) - x_r(t), y(t) - y_r(t))$ at zero, with $(x(t), y(t))$ denoting the coordinates of point P in $\mathcal{F}_0$ at time t. The problem may also be formulated as one of controlling the vehicle in order to track a reference vehicle whose trajectory is given by $t \longmapsto (x_r(t), y_r(t))$. Note that perfect tracking is achievable only if the reference trajectory is feasible for the physical vehicle, and that a trajectory which is feasible for a unicycle-type vehicle is not necessarily feasible for a car-like vehicle. Also, in addition to monitoring the position $(x(t), y(t))$ of the robot, one may be willing to control the chassis orientation $\theta(t)$ at a desired reference value $\theta_r(t)$ associated with the orientation of the reference vehicle. For a nonholonomic unicycle-type robot, a reference trajectory $(x_r(t), y_r(t), \theta_r(t))$ is feasible if it is produced by a reference vehicle which has the same kinematic limitations as the physical robot. For instance, most trajectories produced by an omnidirectional vehicle (omnibile vehicle in the terminology of Chap. 24) are not feasible for a nonholonomic mobile robot. However, nonfeasibility does not imply that the reference trajectory cannot be tracked in an approximate manner, i. e., with small (although nonzero) tracking errors.

49.1.3 Stabilization of Fixed Postures

Let $\mathcal{F}_1$ denote a frame attached to the robot chassis. In this chapter, we call a robot posture (or situation) the association of the position of a point P located on the robot chassis with the orientation $\theta(t)$ of $\mathcal{F}_1$ with respect to a fixed frame $\mathcal{F}_0$ in the plane of motion. For

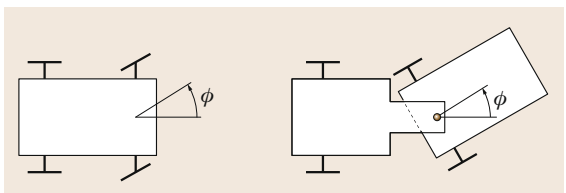


Fig. 49.3 Analogy car/unicycle with trailer

Avidbots Ex1016
Page 1263 of 2154

this last problem, the objective is to stabilize at zero the posture vector $\xi(t) = (x(t), y(t), \theta(t))$, with $(x(t), y(t))$ denoting the position of P expressed in $\mathcal{F}_0$. Although a fixed desired (or reference) posture is obviously a particular case of a feasible trajectory, this problem cannot be solved by classical control methods.

49.2 Control Models

49.2.1 Kinematics Versus Dynamics

In order to proceed with the control of mobile robots, a model describing its motion has to be derived. Several levels of motion equations may be derived, pending on the targeted application (expected velocity, terrain configuration, etc.). Chapter 24 provides a general configuration dynamic model for WMRs. Its particularization to the case of unicycle-type and car-like mobile robots gives

$$\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{u}} + \boldsymbol{F}(\boldsymbol{q}, \boldsymbol{u})\boldsymbol{u} = \boldsymbol{\Gamma}(\phi)\boldsymbol{\tau} \,, \tag{49.1}$$

with $\boldsymbol{q}$ denoting a robot's configuration vector, $\boldsymbol{u}$ a vector of independent velocity variables associated with the robot's degrees of freedom, $\mathbf{H}(\boldsymbol{q})$ a reduced inertia matrix (which is invertible for any $\boldsymbol{q}$), $\boldsymbol{F}(\boldsymbol{q}, \boldsymbol{u})\boldsymbol{u}$ a vector of forces combining the contribution of Coriolis and wheel–ground contact forces, ϕ the orientation angle of the car's steering wheel, $\boldsymbol{\Gamma}$ an invertible control matrix (which is constant in the case of a unicycle-type vehicle), and $\boldsymbol{\tau}$ a vector of independent motor torques (whose dimension is equal to the number of degrees of freedom in the case of full actuation, i. e., equal to two for the vehicles considered herein). In the case of a unicycle-type vehicle, a configuration vector is composed of the components of the chassis posture vector $\boldsymbol{\xi}$ and the orientation angles of the castor wheels (with respect to the chassis). In the case of a car-like vehicle, a configuration vector is composed of the components of $\boldsymbol{\xi}$ and the steering wheel angle ϕ. To be complete, this dynamic model must be complemented with kinematic equations in the form

$$\dot{\boldsymbol{q}} = \mathbf{S}(\boldsymbol{q})\boldsymbol{u} \,, \tag{49.2}$$

from which one can extract a reduced kinematic model

$$\dot{\boldsymbol{z}} = \mathbf{B}(\boldsymbol{z})\boldsymbol{u} \,, \tag{49.3}$$

with $\boldsymbol{z} = \boldsymbol{\xi}$, in the case of a unicycle-type vehicle, and $\boldsymbol{z} = (\boldsymbol{\xi}, \phi)$ in the case of a car-like vehicle. In the automatic control terminology, the complete dynamic model (49.1) and (49.2) forms a *control system* which can be written as

$$\dot{\boldsymbol{X}} = f(\boldsymbol{X}, \boldsymbol{\tau}) \quad \text{with} \quad \boldsymbol{X} = (\boldsymbol{q}, \boldsymbol{u})$$

denoting the state vector of this system, and $\boldsymbol{\tau}$ the vector of control inputs. The kinematic models (49.2) and (49.3) are also control systems with respective state vectors $\boldsymbol{q}$ and $\boldsymbol{z}$ and control vector $\boldsymbol{u}$. Any of these models can be used for control design and analysis purposes. In the remainder of this chapter, we have chosen to work with the kinematic model (49.3). By analogy with the motion control of manipulator arms, this boils down to using a model with velocity control inputs, rather than a model with torque control inputs. This is justified when dynamic effects are not preponderant and low-level velocity control loops on the wheels motor are powerful enough to ensure good velocity tracking. In Sect. 49.5, however, when nonideal wheel-ground contact is addressed, dynamics is considered.

49.2.2 Modeling in an Absolute Frame

For the unicycle-type mobile robot, the kinematic model (49.3) used from now on is

$$\begin{cases} \dot{x} = u_1 \cos\theta \,, \\ \dot{y} = u_1 \sin\theta \,, \\ \dot{\theta} = u_2 \,, \end{cases} \tag{49.4}$$

where (x, y) represents the coordinates of the point $\mathrm{P_m}$ located at mid-distance of the actuated wheels, and the angle θ characterizes the robot's chassis orientation (Fig. 49.4). In this equation, u_1 represents the intensity of the vehicle's longitudinal velocity, and u_2 is the chassis instantaneous velocity of rotation. The variables u_1 and u_2 are themselves related to the angular velocity of the actuated wheels via the one-to-one relations

$$u_1 = \frac{r}{2}(\dot{\psi}_\mathrm{r} + \dot{\psi}_\ell) \,,$$
$$u_2 = \frac{r}{2R}(\dot{\psi}_\mathrm{r} - \dot{\psi}_\ell) \,,$$

with r the wheels' radius, R the distance between the two actuated wheels, and $\dot{\psi}_\mathrm{r}$ (resp. $\dot{\psi}_\ell$) the angular velocity of the right (resp. left) rear wheel.

For the car-like mobile robot, the kinematic model (49.3) used from now on is

$$\begin{cases} \dot{x} = u_1 \cos\theta \,, \\ \dot{y} = u_1 \sin\theta \,, \\ \dot{\theta} = \frac{u_1}{L} \tan\phi \,, \\ \dot{\phi} = u_2 \,, \end{cases} \tag{49.5}$$

where ϕ represents the vehicle's steering wheel angle, and L is the distance between the rear and front wheels' axles. In all forthcoming simulations, L is set equal to 1.2 m.

49.2.3 Modeling in a Frénet Frame

The object of this subsection is to generalize the previous kinematic equations when the reference frame is a Frénet frame. This generalization will be used later on when addressing the path following problem.

Let us consider a curve C in the plane of motion, as illustrated on Fig. 49.5, and let us define three frames $\mathcal{F}_0$, $\mathcal{F}_m$, and $\mathcal{F}_s$, as follows. $\mathcal{F}_0 = \{0, \boldsymbol{i}, \boldsymbol{j}\}$ is a fixed frame, $\mathcal{F}_m = \{P_m, \boldsymbol{i}_m, \boldsymbol{j}_m\}$ is a frame attached to the mobile robot with its origin – the point P_m – located on the rear wheels axle, at the mid-distance of the wheels, and $\mathcal{F}_s = \{P_s, \boldsymbol{i}_s, \boldsymbol{j}_s\}$, which is indexed by the curve's curvilinear abscissa s, is such that the unit vector $\boldsymbol{i}_s$ tangents C. Consider now a point P attached to the robot chassis, and let (l_1, l_2) denote the coordinates of P expressed in the basis of $\mathcal{F}_m$. To determine the equations of motion of P with respect to the curve C let us introduce three variables s, d, and θ_e, defined as follows:

- s is the curvilinear abscissa at the point P_s obtained by projecting P orthogonally on C. This point exists and is unique if the point P is close enough to the curve. More precisely, it suffices that the distance between P and the curve be smaller than the lower bound of the curve radii. We will assume that this condition is satisfied.
- d is the ordinate of P in the frame $\mathcal{F}_s$; its absolute value is also the distance between P and the curve.
- $\theta_e = \theta - \theta_s$ is the angle characterizing the orientation of the robot chassis with respect to the frame $\mathcal{F}_s$.

We also define the curvature $c(s)$ of C at P_s, i. e., $c(s) = \partial\theta_s / \partial s$.

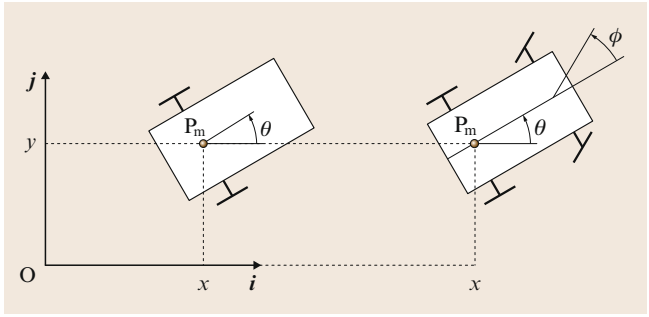


Fig. 49.4 Configuration variables

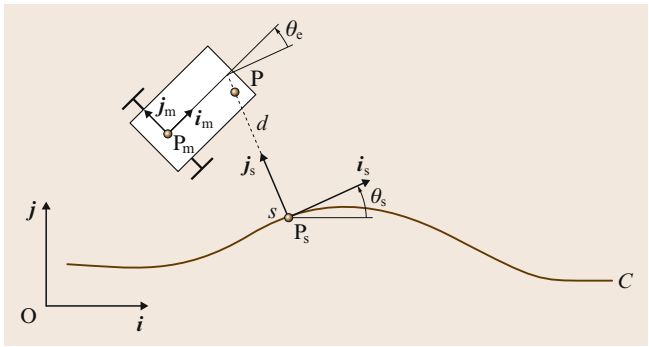


Fig. 49.5 Representation in a Frénet frame

Using these notations one easily deduces from (49.4) the following system of equations ([49.2] for details), which generalizes (49.4)

$$\begin{cases} \dot{s} = \frac{1}{1 - dc(s)} \left[(u_1 - l_2 u_2) \cos\theta_e - l_1 u_2 \sin\theta_e \right] , \\ \dot{d} = (u_1 - l_2 u_2) \sin\theta_e + l_1 u_2 \cos\theta_e \,, \\ \dot{\theta}_e = u_2 - \dot{s} c(s) \,. \end{cases} \tag{49.6}$$

For car-like vehicles, one easily verifies, by using (49.5), that the system (49.6) becomes

$$\begin{cases} \dot{s} = \frac{u_1}{1 - dc(s)} \left[\cos\theta_e - \frac{\tan\phi}{L} (l_2 \cos\theta_e + l_1 \sin\theta_e) \right] , \\ \dot{d} = u_1 \left[\sin\theta_e + \frac{\tan\phi}{L} (l_1 \cos\theta_e - l_2 \sin\theta_e) \right] , \\ \dot{\theta}_e = \frac{u_1}{L} \tan\phi - \dot{s} c(s) \,, \\ \dot{\phi} = u_2 \,. \end{cases} \tag{49.7}$$

Avidbots Ex1016
Page 1265 of 2154

49.3 Adaptation of Control Methods for Holonomic Systems

We address in this section the problems of trajectory stabilization and path following. When we defined these problems in the introduction, we considered a reference point P attached to the robot chassis. It turns out that the choice of this point is important. Indeed, consider for instance the equations (49.6) for a unicycle point P when C is the axis $(O, \boldsymbol{i})$. Then, $s = x_P$, $d = y_P$, and $\theta_e = \theta$ represent the robot's posture with respect to the fixed reference frame $\mathcal{F}_0$. There are two possible cases depending on whether P is, or is not, located on the actuated wheels axle. Let us consider the first case, for which $l_1 = 0$. From the first two equations of (49.6), one has

$$\dot{x}_P = (u_1 - l_2 u_2)\cos\theta\,, \quad \dot{y}_P = (u_1 - l_2 u_2)\sin\theta\,.$$

These relations indicate that P can move only in the direction of the vector $(\cos\theta, \sin\theta)$. This is a direct consequence of the nonholonomy constraint to which the vehicle is subjected. Now, if P is not located on the wheels axle, then

$$\begin{pmatrix}\dot{x}_P\\ \dot{y}_P\end{pmatrix} = \begin{pmatrix}\cos\theta & -l_1\sin\theta\\ \sin\theta & l_1\cos\theta\end{pmatrix}\begin{pmatrix}1 & -l_2\\ 0 & 1\end{pmatrix}\begin{pmatrix}u_1\\ u_2\end{pmatrix}. \tag{49.8}$$

The fact that the two square matrices in the right-hand side of this equality are invertible indicates that $\dot{x}_P$ and $\dot{y}_P$ can take any values, and thus that the motion of P is not constrained. By analogy with holonomic manipulator arms, this means that P may be seen as the extremity of a two-degree-of-freedom (2-DOF) manipulator, and thus that it can be controlled by applying the same control laws as those used for manipulators. In this section, we assume that the point P, used to characterize the robot's position, is chosen away from the rear wheels axle. In this case we will see that the problems of trajectory stabilization and path following can be solved very simply. However, as shown in the subsequent section, choosing P on the wheels axle may also be of interest in order to better control the vehicle's orientation.

49.3.1 Stabilization of Trajectories for a Nonconstrained Point

Unicycle

Consider a differentiable reference trajectory $t \longmapsto (x_r(t), y_r(t))$ in the plane. Let $e = (x_P - x_r, y_P - y_r)$ denote the tracking error in position. The control objective is to asymptotically stabilize this error at zero. In view of (49.8), the error equations are

$$\dot{e} = \begin{pmatrix}\cos\theta & -l_1\sin\theta\\ \sin\theta & l_1\cos\theta\end{pmatrix}\begin{pmatrix}u_1 - l_2 u_2\\ u_2\end{pmatrix} - \begin{pmatrix}\dot{x}_r\\ \dot{y}_r\end{pmatrix}. \tag{49.9}$$

Introducing new control variables (v_1, v_2) defined by

$$\begin{pmatrix}v_1\\ v_2\end{pmatrix} = \begin{pmatrix}\cos\theta & -l_1\sin\theta\\ \sin\theta & l_1\cos\theta\end{pmatrix}\begin{pmatrix}u_1 - l_2 u_2\\ u_2\end{pmatrix}, \tag{49.10}$$

the equations (49.9) become simply

$$\dot{e} = \begin{pmatrix}v_1\\ v_2\end{pmatrix} - \begin{pmatrix}\dot{x}_r\\ \dot{y}_r\end{pmatrix}.$$

The classical techniques of stabilization for linear systems can then be used. For instance, one may consider a proportional feedback control with precompensation such as

$$v_1 = \dot{x}_r - k_1 e_1 = \dot{x}_r - k_1(x_P - x_r)\,, \quad (k_1 > 0)\,,$$
$$v_2 = \dot{y}_r - k_2 e_2 = \dot{y}_r - k_2(y_P - y_r)\,, \quad (k_2 > 0)\,,$$

which yields the closed-loop equation $\dot{e} = -Ke$. Of course, this control can be rewritten for the initial control variables $\boldsymbol{u}$, since the mapping $(u_1, u_2) \longmapsto (v_1, v_2)$ is bijective.

Car

This technique extends to car-like vehicles (and also to trailer systems) by choosing a point P attached to the steering wheel frame and not located on the steering wheel axle.

49.3.2 Path Following with no Orientation Control

Unicycle

Let us adopt the notation of Fig. 49.5 to address the problem of following a path associated with a curve C in the plane. The control objective is to stabilize the distance d at zero. From (49.6), one has

$$\dot{d} = u_1\sin\theta_e + u_2(-l_2\sin\theta_e + l_1\cos\theta_e)\,. \tag{49.11}$$

Recall that in this case the vehicle's longitudinal velocity u_1 is either imposed or prespecified. We will assume that the product $l_1 u_1$ is positive, i. e., the position of the point P with respect to the actuated wheels axle is chosen in relation to the sign of u_1. This assumption will

be removed in Sect. 49.4. To simplify, we will also assume that $l_2 = 0$, i. e., the point P is located on the axis $(P_m, \boldsymbol{i}_m)$. Let us then consider the following feedback control law

$$u_2 = -\frac{u_1 \tan\theta_e}{l_1} - \frac{u_1}{\cos\theta_e} k(d, \theta_e) d \,, \qquad (49.12)$$

with k a continuous, strictly positive, function on $\mathbb{R} \times (-\pi/2, \pi/2)$ such that $k(d, \pm\pi/2) = 0$. Since $l_2 = 0$, applying the control (49.12) to (49.11) gives

$$\dot{d} = -l_1 u_1 k(d, \theta_e) d \,,$$

which suggests that the control law (49.12) ensures convergence of d to zero under adequate conditions on l_1, u_1, and k. This is made precise in the following result ([49.2] for details).

Proposition 49.1

Consider the path following problem for a unicycle-type mobile robot with

A. A strictly positive, or strictly negative, longitudinal velocity u_1.

B. A reference point P of coordinates $(l_1, 0)$ in the vehicle's chassis frame, with $l_1 u_1 > 0$.

Let k denote a continuous function, strictly positive on $\mathbb{R} \times (-\pi/2, \pi/2)$, and such that $k(d, \pm\pi/2) = 0$ for every d (for instance, $k(d, \theta_e) = k_0 \cos\theta_e$). Then, for any initial conditions $(s(0), d(0), \theta_e(0))$ such that

$$\theta_e(0) \in (-\pi/2, \pi/2) \,, \quad \frac{l_1 c_{\max}}{1 - |d(0)| c_{\max}} < 1 \,,$$

with $c_{\max} = \max_s |c(s)|$, the feedback control (49.12) makes the distance $|d|$ between P and the curve non-increasing, and makes it converge to zero if

$$\int_0^t |u_1(s)| \mathrm{d}s \longrightarrow +\infty \quad \text{when } t \longrightarrow +\infty \,.$$

Car

This control technique also applies to car (or trailer systems) by considering a point P attached to the steering wheel frame, with u_1 positive.

49.4 Methods Specific to Nonholonomic Systems

The control methods recalled in the previous section are simple and sufficient for many applications. Their main limitation comes from the fact that the robot's orientation is not actively controlled. This becomes a problem when the application requires manœuvers (i. e., when the vehicle's velocity u_1 has to become negative). This issue is especially critical for trailer systems (including car-like vehicles). Indeed, when the longitudinal velocity is positive, the leader vehicle has a *pulling* action which tends to align the followers along the curve. In the other case, the leader has a pushing action which tends to misalign them, thus leading to collisions between the vehicles' components (the jackknife effect, see Fig. 49.6 for illustration). In order to remove this constraint on the sign of the longitudinal velocity, the control has to be designed so that all orientation angles are actively stabilized. An indirect way to do this consists in choosing the point P on the actuated wheels axle, at the mid-distance of the wheels, for instance. In this case, the nonholonomy constraints intervene much more explicitly, and the control can no longer be obtained by applying the techniques used for holonomic manipulators.

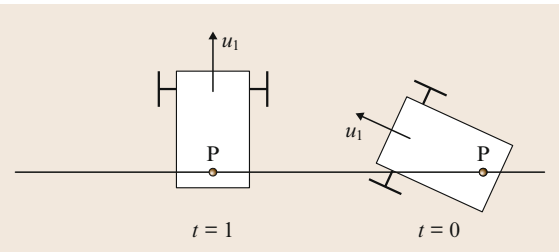


Fig. 49.6 Path-following instability with reverse longitudinal velocity

This section is organized as follows. First, the modeling equations with respect to a Frénet frame are recast into a canonical form called the chained form. From there, a solution to the path-following problem with active stabilization of the vehicle's orientation is worked out. The problem of (feasible) trajectory stabilization is also revisited with the complementary objective of controlling the vehicle's orientation. The asymptotic stabilization of fixed postures is then addressed. Finally, some comments on the limitations of the proposed control strategies, in relation to the objective of asymptotic stabilization, serve to motivate and introduce a new control approach developed in the subsequent section.

Avidbots Ex1016
Page 1267 of 2154

49.4.1 Transformation of Kinematic Models into the Chained Form

The chapter dedicated to path planning shows how the kinematic equations of the mobile robots here considered (unicycle-type, car-like, with trailers) can be transformed into the chain form via a change of state and control variables. In particular, the equations of a unicycle (49.4), and those of a car (49.5), can be transformed into a three-dimensional and a four-dimensional chained system, respectively. Those of a unicycle-type vehicle with N trailers yield a chained system of dimension $N+3$ when the trailers are hooked to each other in a specific way. As shown below, this transformation can be generalized to the kinematic models derived with respect to a Frénet frame. The result will be given only for the unicycle and car cases (49.6) and (49.7), but it also holds for trailer systems. The reference point P is now chosen at the mid-distance of the vehicle's rear wheels (or at the mid-distance of the wheels of the last trailer, when trailers are involved).

Let us start with the unicycle case. Under the assumption that P corresponds to the origin of $\mathcal{F}_m$, one has $l_1 = l_2 = 0$ so that the system (49.6) simplifies to

$$\begin{cases} \dot{s} = \dfrac{u_1}{1-dc(s)} \cos\theta_e \,, \\ \dot{d} = u_1 \sin\theta_e \,, \\ \dot{\theta}_e = u_2 - \dot{s}c(s) \,. \end{cases} \tag{49.13}$$

Let us determine a change of coordinates and control variables $(s, d, \theta_e, u_1, u_2) \longmapsto (z_1, z_2, z_3, v_1, v_2)$ allowing to (locally) transform (49.13) into the three-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = v_2 \,. \end{cases} \tag{49.14}$$

By first setting

$$z_1 = s \,, \quad v_1 = \dot{s} = \frac{u_1}{1-dc(s)} \cos\theta_e \,,$$

we already obtain $\dot{z}_1 = v_1$. This implies that

$$\begin{aligned} \dot{d} &= u_1 \sin\theta_e = \frac{u_1}{1-dc(s)} \cos\theta_e [1-dc(s)] \tan\theta_e \\ &= v_1 [1-dc(s)] \tan\theta_e \,. \end{aligned}$$

We then set $z_2 = d$ and $z_3 = [1-dc(s)]\tan\theta_e$, so that the above equation becomes $\dot{z}_2 = v_1 z_3$. Finally, we define

$$\begin{aligned} v_2 &= \dot{z}_3 \\ &= \left[-\dot{d}c(s) - d\frac{\partial c}{\partial s}\dot{s} \right] \tan\theta_e \\ &\quad + [1-dc(s)](1+\tan^2\theta_e)\dot{\theta}_e \,. \end{aligned}$$

The equations (49.14) are satisfied with the variables z_i and v_i so defined.

From this construction it is simple to verify that the mapping $(s, d, \theta_e) \longmapsto z$ is a *local* change of coordinates defined on $\mathbb{R}^2 \times (-\pi/2, \pi/2)$ (to be more rigorous, one should also take the constraint $|d| < 1/c(s)$ into account). Let us finally remark that the change of control variables involves the derivative $(\partial c/\partial s)$ of the path's curvature (whose knowledge is thus needed for the calculations). In summary, we have shown that the change of coordinates and of control variables $(s, d, \theta_e, u_1, u_2) \longmapsto (z_1, z_2, z_3, v_1, v_2)$ defined by

$$\begin{aligned} (z_1, z_2, z_3) &= (s, d, [1-dc(s)]\tan\theta_e) \\ (v_1, v_2) &= (\dot{z}_1, \dot{z}_3) \end{aligned}$$

transforms the model (49.13) of a unicycle-type vehicle into a three-dimensional (3-D) chained system. One can similarly transform the car's equations into a four-dimensional (4-D) chained system, although the calculations are slightly more cumbersome. More precisely, the change of coordinates and control variables $(s, d, \theta_e, \phi, u_1, u_2) \longmapsto (z_1, z_2, z_3, z_4, v_1, v_2)$ defined by

$$\begin{aligned} (z_1, z_2, z_3, z_4) = \Big\{ & s, d, [1-dc(s)]\tan\theta_e, \\ & -c(s)[1-dc(s)]\left(1+2\tan^2\theta_e\right) \\ & -d\frac{\partial c}{\partial s}\tan\theta_e \\ & +[1-dc(s)]^2 \frac{\tan\phi}{L}\frac{1+\tan^2\theta_e}{\cos\theta_e} \Big\} \,, \\ (v_1, v_2) = (\dot{z}_1, \dot{z}_4) \end{aligned}$$

transforms the model (49.7) of a car-like vehicle (with $l_1 = l_2 = 0$) into a four-dimensional chained system.

49.4.2 Tracking of a Reference Vehicle with the Same Kinematics

Let us first consider the problem of tracking, in both position and orientation, a reference vehicle (Fig. 49.7). For simplicity, we choose P as the origin P_m of the robot's chassis frame $\mathcal{F}_m$.

Although the terminology is rather loose, the *tracking problem* is usually associated, in the control literature, with the problem of *asymptotically stabilizing* the reference trajectory. In this case, a necessary condition for the existence of a control solution is that the reference is feasible. Feasible trajectories $t \longmapsto (x_r(t), y_r(t), \theta_r(t))$ are smooth time functions which are solution to the robot's kinematic model for some specific control input $t \longmapsto \boldsymbol{u}_r(t) = (u_{1,r}(t), u_{2,r}(t))^T$, called the *reference control*. For a unicycle-type robot for example, this means in view of (49.4) that

$$\begin{cases} \dot{x}_r = u_{1,r} \cos\theta_r \,, \\ \dot{y}_r = u_{1,r} \sin\theta_r \,, \\ \dot{\theta}_r = u_{2,r} \,. \end{cases} \tag{49.15}$$

In other words, feasible reference trajectories correspond to the motion of a reference frame $\mathcal{F}_r = \{P_r, \boldsymbol{i}_r, \boldsymbol{j}_r\}$ rigidly attached to a reference unicycle-type robot, with P_r (alike $P = P_m$) located at the mid-distance of the actuated wheels (Fig. 49.7). From there, the problem is to determine a feedback control which asymptotically stabilizes the tracking error $(x - x_r, y - y_r, \theta - \theta_r)$ at zero, with (x_r, y_r) being the coordinates of P_r in $\mathcal{F}_0$, and θ_r the oriented angles between $\boldsymbol{i}$ and $\boldsymbol{i}_r$. One can proceed as in the path-following case, first by establishing the error equations with respect to the frame $\mathcal{F}_r$, then by transforming these equations into the chain form via a change of variables like the one used to transform the kinematic equations of a mobile robot into a chained system, and finally by designing stabilizing control laws for the transformed system.

Expressing the tracking error in position $(x - x_r, y - y_r)$ with respect to the frame $\mathcal{F}_r$ gives the vector (Fig. 49.7)

$$\begin{pmatrix} x_e \\ y_e \end{pmatrix} = \begin{pmatrix} \cos\theta_r & \sin\theta_r \\ -\sin\theta_r & \cos\theta_r \end{pmatrix} \begin{pmatrix} x - x_r \\ y - y_r \end{pmatrix} . \tag{49.16}$$

Calculating the time derivative of this vector yields

$$\begin{pmatrix} \dot{x}_e \\ \dot{y}_e \end{pmatrix} = \begin{pmatrix} u_{2,r} y_e + u_1 \cos(\theta - \theta_r) - u_{1,r} \\ -u_{2,r} x_e + u_1 \sin(\theta - \theta_r) \end{pmatrix} .$$

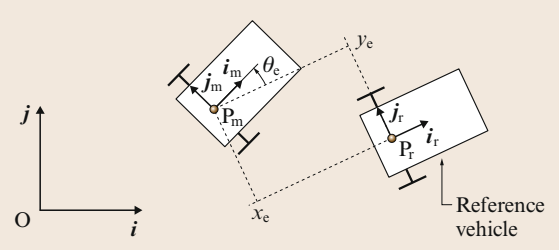


Fig. 49.7 Reference vehicle and error coordinates

By denoting $\theta_e = \theta - \theta_r$, the orientation error between the frames $\mathcal{F}_m$ and $\mathcal{F}_r$, we obtain

$$\begin{cases} \dot{x}_e = u_{2,r} y_e + u_1 \cos\theta_e - u_{1,r} \,, \\ \dot{y}_e = -u_{2,r} x_e + u_1 \sin\theta_e \,, \\ \dot{\theta}_e = u_2 - u_{2,r} \,. \end{cases} \tag{49.17}$$

To determine a control (u_1, u_2) which asymptotically stabilizes the error (x_e, y_e, θ_e) at zero, let us consider the following change of coordinates and control variables

$$(x_e, y_e, \theta_e, u_1, u_2) \longmapsto (z_1, z_2, z_3, w_1, w_2) \,,$$

defined by

$$\begin{aligned} z_1 &= x_e \,, \\ z_2 &= y_e \,, \\ z_3 &= \tan\theta_e \,, \\ w_1 &= u_1 \cos\theta_e - u_{1,r} \,, \\ w_2 &= \frac{u_2 - u_{2,r}}{\cos^2\theta_e} \,. \end{aligned} \tag{49.18}$$

Note that, around zero, this mapping is only defined when $\theta_e \in (-\pi/2, \pi/2)$. In other words, the orientation error between the physical robot and the reference robot has to be smaller than $\pi/2$.

It is immediate to verify that, in the new variables, the system (49.17) can be written as

$$\begin{cases} \dot{z}_1 = u_{2,r} z_2 + w_1 \,, \\ \dot{z}_2 = -u_{2,r} z_1 + u_{1,r} z_3 + w_1 z_3 \,, \\ \dot{z}_3 = w_2 \,. \end{cases} \tag{49.19}$$

We remark that the last term in each of the above three equations corresponds to the one of a chained system. From here, it can be shown that the feedback control law

$$\begin{cases} w_1 = -k_1 |u_{1,r}| (z_1 + z_2 z_3) \quad (k_1 > 0) \,, \\ w_2 = -k_2 u_{1,r} z_2 - k_3 |u_{1,r}| z_3 \,, \quad (k_2, k_3 > 0) \,, \end{cases} \tag{49.20}$$

makes the origin of system (49.19) globally asymptotically stable provided essentially that $u_{1,r}$ does not tend to zero as t tends to infinity. A precise stability statement and associated proof are provided in [49.2], together with complementary details about the gain tuning. Finally, the velocity control inputs u_1 and u_2 can be deduced from w_1 and w_2 by using the two last lines of (49.18).

Avidbots Ex1016
Page 1269 of 2154

Generalization to a Car-Like Vehicle

The previous method extends to the car case. We provide below the main steps of this extension. Consider the car's kinematic model (49.5) and a reference vehicle satisfying the same kinematic equations with all variables indexed by the subscript r. We assume that there exists $\delta \in (0, \pi/2)$ such that the reference steering angle ϕ_r belongs to the interval $[-\delta, \delta]$. By defining x_e, y_e, and θ_e as in the unicycle case, and by setting $\phi_e = \phi - \phi_r$, one obtains the following system

$$\begin{cases} \dot{x}_e = \left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) y_e + u_1\cos\theta_e - u_{1,r}\,, \\ \dot{y}_e = -\left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) x_e + u_1\sin\theta_e\,, \\ \dot{\theta}_e = \dfrac{u_1}{L}\tan\phi - \dfrac{u_{1,r}}{L}\tan\phi_r\,, \\ \dot{\phi}_e = u_2 - u_{2,r}\,. \end{cases} \tag{49.21}$$

Let us introduce the following new state variables

$$\begin{cases} z_1 = x_e\,, \\ z_2 = y_e\,, \\ z_3 = \tan\theta_e\,, \\ z_4 = \dfrac{\tan\phi - \cos\theta_e\tan\phi_r}{L\cos^3\theta_e} + k_2 y_e\,, (k_2 > 0)\,. \end{cases}$$

We note that for any $\phi_r \in (-\pi/2, \pi/2)$, the mapping $(x_e, y_e, \theta_e, \phi) \longmapsto z$ defines a diffeomorphism between $\mathbb{R}^2 \times (-\pi/2, \pi/2)^2$ and $\mathbb{R}^4$. Introduce now the new control variables

$$\begin{cases} w_1 = u_1\cos\theta_e - u_{1,r}\,, \\ w_2 = \dot{z}_4 = k_2\dot{y}_e + \left(\dfrac{3\tan\phi}{\cos\theta_e} - 2\tan\phi_r\right)\dfrac{\sin\theta_e}{L\cos^3\theta_e}\dot{\theta}_e \\ \quad - \dfrac{u_{2,r}}{L\cos^2\phi_r\cos^2\theta_e} + \dfrac{u_2}{L\cos^2\phi\cos^3\theta_e}\,. \end{cases} \tag{49.22}$$

One shows that $(u_1, u_2) \longmapsto (w_1, w_2)$ defines a change of variables for θ_e, ϕ, and ϕ_r, inside the interval $(-\pi/2, \pi/2)$. These changes of state and control variables transform the system (49.21) into

$$\begin{cases} \dot{z}_1 = \left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) z_2 + w_1\,, \\ \dot{z}_2 = -\left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) z_1 + u_{1,r} z_3 + w_1 z_3\,, \\ \dot{z}_3 = -k_2 u_{1,r} z_2 + u_{1,r} z_4 \\ \quad + w_1\left[z_4 - k_2 z_2 + (1+z_3^2)\dfrac{\tan\phi_r}{L}\right]\,, \\ \dot{z}_4 = w_2\,. \end{cases} \tag{49.23}$$

From here, one can deduce that the control law w_1 and w_2 defined as

$$\begin{cases} w_1 = -k_1|u_{1,r}| \\ \qquad \times \left\{z_1 + \dfrac{z_3}{k_2}\left[z_4 + (1+z_3^2)\dfrac{\tan\phi_r}{L}\right]\right\}\,, \\ w_2 = -k_3 u_{1,r} z_3 - k_4|u_{1,r}| z_4\,, \end{cases} \tag{49.24}$$

with $k_{1,2,3,4}$ denoting positive numbers, makes the origin of system (49.23) globally asymptotically stable provided essentially that $u_{1,r}$ does not tend to zero as t tends to infinity.

Simulation Results

The simulation shown on Fig. 49.8 and VIDEO 181 illustrates this control scheme. The gain parameters k_i have been chosen as $(k_1, k_2, k_3, k_4) = (1, 1, 3, 3)$. The initial configuration of the reference vehicle (i. e., at $t = 0$), which is represented in Fig. 49.8a by dashed lines, is $(x_r, y_r, \theta_r)(0) = (0, 0, 0)$. The reference control $\boldsymbol{u}_r$ is defined by (49.25). The initial configuration of the controlled robot, represented in the figure by plain lines, is $(x, y, \theta)(0) = (0, -1.5, 0)$. The configurations at time $t = 10$, 20, and 30, are also represented on the figure. Due to the fast convergence of the tracking error to zero (see the time evolution of the components x_e, y_e, θ_e of the tracking error in Fig. 49.8b), one can basically consider that the configurations of both vehicles coincide after time $t = 10$.

$$u_r(t) = \begin{cases} (1,0)^T\,, & \text{if } t \in [0, 10]\,, \\ \{-1, 0.5\cos[2\pi(t-10)/5]\}^T\,, \\ \text{if } t \in [10, 20]\,, \\ (1,0)^T\,, & \text{if } t \in [20, 30]\,. \end{cases} \tag{49.25}$$

49.4.3 Path Following with Orientation Control

We reconsider the path-following problem with the reference point P now located on the actuated wheels axle, at the mid-distance of the wheels. The objective is to synthesize a control law which allows the vehicle to follow the path in a stable manner, independently of the sign of the longitudinal velocity.

Unicycle Case

We have seen in Sect. 49.4.1 how to transform kinematic equations with respect to a Frénet frame into the

three-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = v_2 \,. \end{cases} \tag{49.26}$$

Recall that $(z_1, z_2, z_3) = (s, d, (1 - dc(s)) \tan \theta_e)$ and that $v_1 = u_1/(1 - dc(s)) \cos \theta_e$. The objective is to determine a control law which asymptotically stabilizes $(d = 0, \theta_e = 0)$ and also ensures that the constraint on the distance d to the path (i. e., $|dc(s)| < 1$) is satisfied along the trajectories of the controlled system. For the control law, a first possibility consists in considering a proportional feedback like

$$v_2 = -v_1 k_2 z_2 - |v_1| k_3 z_3 \,, \quad (k_2, k_3 > 0) \,. \tag{49.27}$$

It is straightforward to verify that the origin of the closed-loop subsystem

$$\begin{cases} \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = -v_1 k_2 z_2 - |v_1| k_3 z_3 \end{cases} \tag{49.28}$$

is asymptotically stable when v_1 is constant, either positive or negative. Since u_1 (not v_1) is the intensity of the vehicle's longitudinal velocity, one would rather establish stability conditions which depend on u_1. As detailed in [49.2], the control law (49.27) in fact makes the origin of system (49.28) asymptotically stable under the main condition that u_1 does not tend to zero a t tends to infinity. Note that the sign of u_1 now does not need to be constant. As for the constraint $|dc(s)| < 1$, it is satisfied along any solution to the controlled system provided that

$$z_2^2(0) + \frac{1}{k_2} z_3^2(0) < \frac{1}{c_{\max}^2} \,,$$

with $c_{\max} = \max_s |c(s)|$.

From a practical point of view it can be useful to complement the control action with an integral term. More precisely, let us define a variable z_0 by $\dot{z}_0 = v_1 z_2$ with $z_0(0) = 0$. The control (49.27) can be modified as follows

$$\begin{aligned} v_2 &= -|v_1| k_0 z_0 - v_1 k_2 z_2 - |v_1| k_3 z_3 \,, \\ &= -|v_1| k_0 \int_0^t v_1 z_2 - v_1 k_2 z_2 - |v_1| k_3 z_3 \,, \\ &(k_0, k_2, k_3 > 0, k_0 < k_2 k_3) \end{aligned} \tag{49.29}$$

leading to similar stability results as before. The constraint $|dc(s)| < 1$ is now satisfied along any solution to the controlled system provided that

$$z_2^2(0) + \frac{1}{k_2 - \frac{k_0}{k_3}} z_3^2(0) < \frac{1}{c_{\max}^2} \,.$$

Generalization

One of the assets of this type of approach, besides the simplicity of the control law and little demanding conditions of stability associated with it, is that it can be generalized in a straightforward manner to car-like vehicles and unicycle-type vehicles with trailers. The result is summarized hereafter by considering an n-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \vdots \\ \dot{z}_{n-1} = v_1 z_n \,, \\ \dot{z}_n = v_2 \,, \end{cases} \tag{49.30}$$

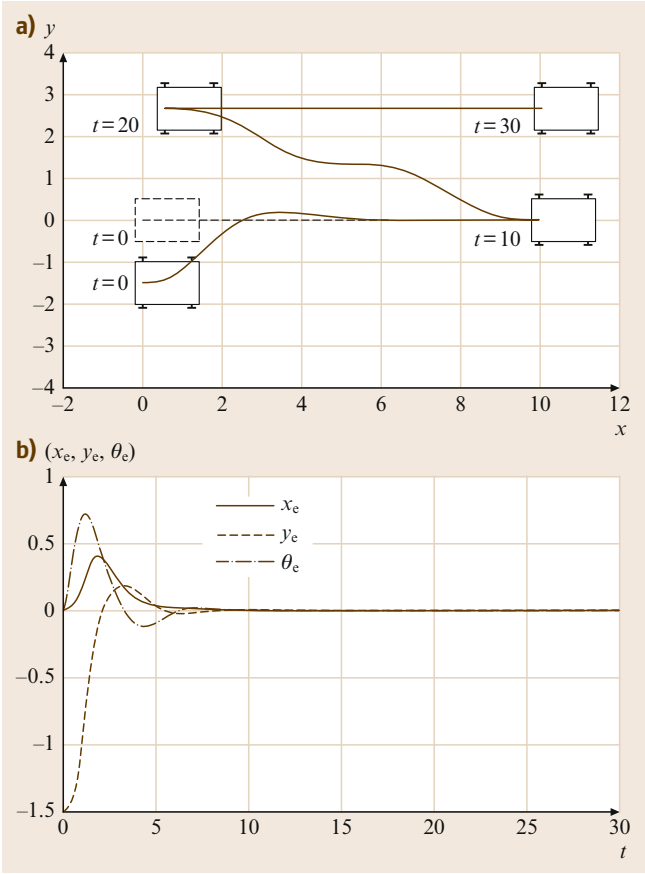


Fig.49.8a,b Tracking of a reference vehicle. **(a)** Cartesian motion; **(b)** error coordinates versus time

with $n \geq 3$. Its proof is a direct extension of the one in the three-dimensional case. The dimension $n = 4$ corresponds to the car case (Sect. 49.4.1). As for a unicycle-type vehicle with N trailers, one has $n = N + 3$. Recall also that, in all cases, z_2 represents the distance d between the path and the point P located at the mid-distance of the rear wheels of the last vehicle. The control objective is then to ensure the convergence of $z_2, \ldots, z_n$ to zero. A solution to this problem, which extends the solution (49.27) proposed for the unicycle, is given by

$$v_2 = -v_1 \sum_{i=2}^{n} \text{sign}(v_1)^{n+1-i} k_i z_i \,. \tag{49.31}$$

where the control gains $k_2, \ldots, k_n$ are chosen such that the polynomial

$$s^{n-1} + k_n s^{n-2} + k_{n-1} s^{n-3} + \ldots + k_3 s + k_2$$

is Hurwitz stable. Satisfaction of the constraint $|dc(s)| < 1$ along any solution to the controlled system can also be guaranteed, upon a condition on $(z_2(0), \cdots, z_n(0))$ specified in [49.2]. The possibility of adding an integral correction term in (49.31) is also addressed in [49.2].

Simulation Results

The simulation results reported in Fig. 49.9 and VIDEO 181 illustrate how this control scheme performs for a car-like vehicle. The reference curve is the circle of radius equal to four, centered at the origin. The robot's longitudinal velocity u_1 is defined by $u_1 = 1$ for $t \in [0, 5]$, and $u_1 = -1$ for $t > 5$. The control gains have been chosen as $(k_2, k_3, k_4) = (1, 3, 3)$. The motion of the car-like robot in the plane is represented in Fig. 49.9a, and its configuration at times $t = 0, 5$, and 25 are also depicted in the figure. The time evolution of the variables z_2, z_3, z_4 (defined at the end of Sect. 49.4.1) is represented in Fig. 49.9b. One can observe that the (discontinuous) change of the longitudinal velocity u_1 at $t = 5$ does not affect the convergence of these variables to zero.

49.4.4 Asymptotic Stabilization of Fixed Postures

We now consider the problem of asymptotic stabilization of a fixed desired (reference) posture (i. e., position and orientation) for the robot chassis. This problem may be seen as a limit case of the trajectory tracking problem. However, none of the feedback controllers proposed previously in this chapter provides a solution to this problem. From the automatic control point of view, the asymptotic stabilization of fixed postures is very different from the problems of path following and trajectory tracking with nonzero longitudinal velocity, much in the same way as a human driver knows, from experience, that parking a car at a precise location involves techniques and skills different from those exercised when cruising on a road. In particular, it cannot be solved by any classical control method for linear systems (or based on linearization). Technically, the underlying general problem is the one of asymptotic stabilization of equilibria of controllable driftless systems with less control inputs than state variables. This problem has motivated numerous studies during the last decade of the last century, from many authors and with various angles of attack, and it has remained a subject of active research at the beginning of this century.

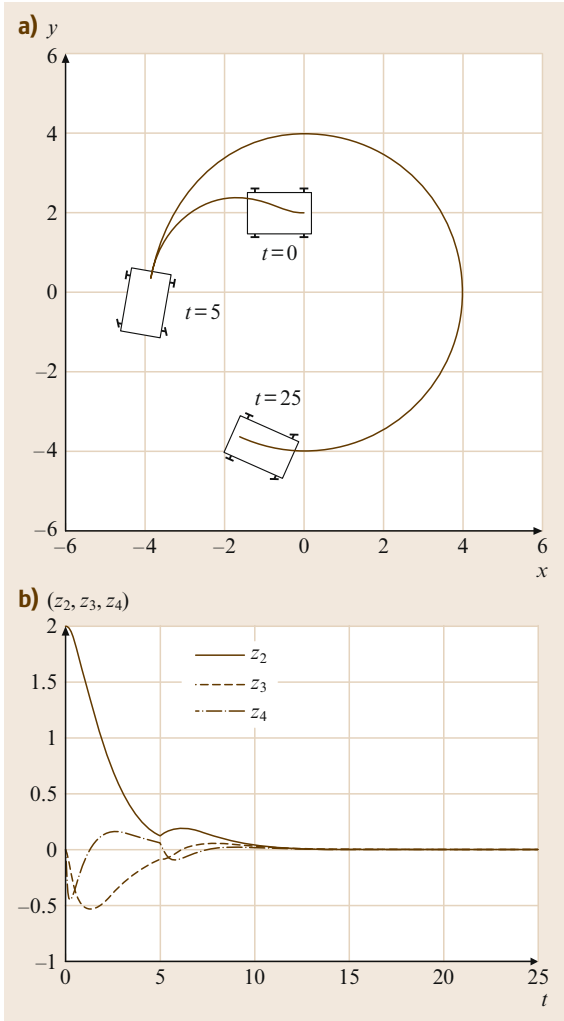


Fig.49.9a,b Path following along a circle. **(a)** Cartesian motion; **(b)** coordinates $z_{2,3,4}$ versus time

Avidbots Ex1016
Page 1272 of 2154

The variety of candidate solutions proposed until now, the mathematical technicalities associated with several of them, together with unsolved difficulties and limitations, particularly (but not only) in terms of robustness (an issue on which we will return), prevent us from attempting to cover the subject exhaustively. Instead, we have opted for a somewhat informal exposition of approaches which have been considered, with the illustration of a few control solutions, without going into technical and mathematical details.

A central aspect of the problem, which triggered much of the subsequent research on the control of nonholonomic systems, is that asymptotic stabilization of equilibria (or fixed points) cannot be achieved by using continuous feedbacks which depend on the state only (i. e., continuous pure-state feedbacks). This is a consequence of an important result due to Brockett in 1983. The original result by Brockett concerned differentiable feedbacks; it has later been extended to the larger set of feedbacks which are only continuous.

Theorem 49.1 (Brockett [49.3])
Consider a control system

$$\dot{x} = f(\boldsymbol{x}, \boldsymbol{u}) \quad (\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{u} \in \mathbb{R}^m) \,,$$

with f a differentiable function and $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$ an equilibrium of this system. A necessary condition for the existence of a continuous feedback control $\boldsymbol{u}(\boldsymbol{x})$ which renders the origin of the closed-loop system

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}(\boldsymbol{x}))$$

asymptotically stable is the local surjectivity of the application $(\boldsymbol{x}, \boldsymbol{u}) \longmapsto f(\boldsymbol{x}, \boldsymbol{u})$. More precisely, the image by f of any neighborhood Ω of $(0, 0)$ in $\mathbb{R}^{n+m}$ must be a neighborhood of 0 in $\mathbb{R}^n$.

This result implies that the equilibria of many controllable (nonlinear) systems are not asymptotically stabilizable by continuous pure-state feedbacks. All nonholonomic WMRs belong to this category of systems. This will be shown in the case of a unicycle-type vehicle; the proof for the other mobile robots is similar. Let us thus consider a unicycle-type vehicle, whose kinematic equations (49.4) can be written as $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ with $\boldsymbol{x} = (x_1, x_2, x_3)^{\mathrm{T}}$, $\boldsymbol{u} = (u_1, u_2)^{\mathrm{T}}$, and $f(\boldsymbol{x}, \boldsymbol{u}) = (u_1 \cos x_3, u_1 \sin x_3, u_2)^{\mathrm{T}}$, and let us show that f is not locally onto in the neighborhood of $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$. To this purpose, take a vector in $\mathbb{R}^3$ of the form $(0, \delta, 0)^{\mathrm{T}}$. It is obvious that the equation $f(\boldsymbol{x}, \boldsymbol{u}) = (0, \delta, 0)^{\mathrm{T}}$ does not have a solution in the neighborhood of $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$ since the first equation, namely $u_1 \cos x_3 = 0$, implies that $u_1 = 0$, so that the second equation cannot have a solution if δ is different from zero.

It is also obvious that the linear approximation about the equilibrium $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$ of the unicycle kinematic equations is not controllable. If it were, it would be possible to (locally) asymptotically stabilize this equilibrium with a linear (thus continuous) state feedback.

Therefore, by application of the above theorem, a unicycle-type mobile robot (like other nonholonomic robots) cannot be asymptotically stabilized at a desired posture (position/orientation) by using a continuous pure-state feedback. This impossibility has motivated the development of other control strategies in order to solve the problem. Three major types of controls have been considered:

1. *Continuous time-varying feedbacks*, which, besides from depending on the state x, depend also on the exogenous time variable (i. e., $\boldsymbol{u}(\boldsymbol{x}, t)$ instead of $\boldsymbol{u}(\boldsymbol{x})$ for classical feedbacks).
2. *Discontinuous feedbacks*, in the classical form $\boldsymbol{u}(\boldsymbol{x})$, except that the function $\boldsymbol{u}$ is not continuous at the equilibrium that one wishes to stabilize.
3. *Hybrid discrete/continuous feedbacks*. Although this class of feedbacks is not defined as precisely as the other two sets of controls, it is mostly composed of time-varying feedbacks, either continuous or discontinuous, such that the part of the control which depends upon the state is only updated periodically, e.g., $\boldsymbol{u}(t) = \bar{\boldsymbol{u}}[\boldsymbol{x}(kT), t]$ for any $t \in [kT, (k+1)T]$, with T denoting a constant period, and $k \in \mathbb{N}$.

We focus hereafter on continuous time-varying feedbacks. Examples of hybrid discrete/continuous feedbacks for unicycle and car-like vehicles are provided in [49.2, Sect 34.4.4], together with associated simulation results. As for discontinuous feedbacks, they involve difficult questions (existence of solutions, mathematical meaning of these solutions, etc.) which complicate their analysis and for which complete answers are not available. Moreover, for most of the discontinuous control strategies described in the literature, the property of stability in the sense of Lyapunov is either not granted or remains an open issue.

The use of time-varying feedbacks for the asymptotic stabilization of a fixed desired equilibrium, for a nonholonomic WMR, in order to circumvent the obstruction pointed out by Brockett's Theorem, was first proposed in [49.4]. Since then, very general results about the stabilization of nonlinear systems by means of time-varying feedbacks have been obtained. For instance, it has been proved that any controllable driftless system can have any of its equilibria asymptotically stabilized with a control of this type [49.5]. This includes the kinematic models of the nonholo-

nomic mobile robots here considered. We will illustrate this approach in the case of unicycle-type and car-like mobile robots modeled by three- and four-dimensional chained systems, respectively. In order to consider the three-dimensional case, let us come back on the results obtained in Sect. 49.4.3 for path following. The control law (49.27), i. e., $v_2 = -v_1 k_2 z_2 - |v_1| k_3 z_3$ applied to the system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = v_2 \,, \end{cases}$$

renders the function

$$V(z) := \frac{1}{2}(z_2^2 + \frac{1}{k_2} z_3^2)$$

nonincreasing along any trajectory of the controlled system, i. e.,

$$\dot{V} = -\frac{k_3}{k_2} |v_1| z_3^2 \,,$$

and ensures the convergence of z_2 and z_3 to zero if, for instance, v_1 does not tend to zero as t tends to infinity. For example, if $v_1(t) = \sin t$, z_2 and z_3 tend to zero, and

$$\begin{aligned} z_1(t) &= z_1(0) + \int_0^t v_1(s) \mathrm{d}s = z_1(0) + \int_0^t \sin s \mathrm{d}s \\ &= z_1(0) + 1 - \cos t \,, \end{aligned}$$

so that $z_1(t)$ oscillates around the mean value $z_1(0) + 1$. To reduce these oscillations, one can multiply v_1 by a factor which depends on the current state. Take, for example, $v_1(z,t) = \|(z_2, z_3)\| \sin t$, that we complement with a stabilizing term like $-k_1 z_1$ with $k_1 > 0$, i. e.,

$$v_1(z,t) = -k_1 z_1 + \|(z_2, z_3)\| \sin t \,.$$

The feedback control so obtained is time-varying and asymptotically stabilizing.

Proposition 49.2

The continuous time-varying feedback

$$\begin{cases} v_1(z,t) = -k_1 z_1 + \alpha \|(z_2, z_3)\| \sin t \,, \\ v_2(z,t) = -v_1(z,t) k_2 z_2 - |v_1(z,t)| k_3 z_3 \,, \end{cases} \tag{49.32}$$

with $\alpha, k_{1,2,3} > 0$, renders the origin of the 3-D chained system globally asymptotically stable [49.6].

The above proposition can be extended to chained systems of arbitrary dimension [49.6]. For the case $n = 4$, which corresponds to the car-like robot, one has the following result.

Proposition 49.3

The continuous time-varying feedback

$$\begin{cases} v_1(z,t) = -k_1 z_1 + \alpha \|(z_2, z_3, z_4)\| \sin t \,, \\ v_2(z,t) = -|v_1(z,t)| k_2 z_2 - v_1(z,t) k_3 z_3 \\ -|v_1(z,t)| k_4 z_4 \,, \end{cases} \tag{49.33}$$

with $\alpha, k_{1,2,3,4} > 0$ chosen such that the polynomial $s^3 + k_4 s^2 + k_3 s + k_2$ is Hurwitz stable, renders the origin of the 4-D chained system globally asymptotically stable [49.6].

Figure 49.10 below illustrates the previous result. For this simulation, the parameters $\alpha, k_{1,2,3,4}$ in the feedback law (49.33) have been chosen as $\alpha = 3$ and $k_{1,2,3,4} = (1.2, 10, 18, 17)$. Figure 49.10a shows the motion of the car-like robot in the plane. The initial configuration, at time $t = 0$, is depicted in plain lines, whereas the desired configuration is shown in dashed lines. The time evolution of the variables x, y, and θ (i. e., the position and orientation variables corresponding to the model (49.5)) is shown in Fig. 49.10b.

A shortcoming of this type of control, very clear from this simulation, is that the system's state converges to zero quite slowly. One can show that the rate of convergence is only polynomial, i. e., it is commensurable with $t^{-\alpha}$ (for some $\alpha \in (0,1)$) for most of the trajectories of the controlled system. This slow rate of convergence is related to the fact that the control function is Lipschitz-continuous with respect to x. It is a characteristics of systems the linear approximation of which is not stabilizable. Indeed, it can be shown that exponential stability (in the classical sense of linear systems) of an equilibrium point of a nonlinear system cannot be obtained with a Lipschitz-continuous feedback when the system's linear approximation at that point is not stabilizable ([49.2, Prop. 34.11] for a precise statement and details). The need of better performance and efficiency, has triggered the development of stabilizing time-varying feedbacks which are continuous, but not Lipschitz-continuous. Examples of such feedbacks, yielding uniform exponential convergence, are given in the following propositions for chained systems of dimension three and four, respectively.

Proposition 49.4

Let $\alpha, k_{1,2,3} > 0$ denote scalars such that the polynomial $p(s) = s^2 + k_3 s + k_2$ is Hurwitz stable. For any integers $p, q \in \mathbb{N}^*$, let $\rho_{p,q}$ denote the function defined on $\mathbb{R}^2$ by

$$\begin{aligned} \forall \bar{z}_2 = (z_2, z_3) \in \mathbb{R}^2, \rho_{p,q}(\bar{z}_2) \\ = \left(|z_2|^{\frac{p}{q+1}} + |z_3|^{\frac{p}{q}} \right)^{\frac{1}{p}} \,. \end{aligned}$$

Avidbots Ex1016
Page 1274 of 2154

Then, there exists $q_0 > 1$ such that, for any $q \geq q_0$ and $p > q + 2$, the continuous state feedback

$$\begin{cases} v_1(z,t) = -k_1(z_1 \sin t - |z_1|) \sin t \\ \quad + \alpha \rho_{p,q}(\bar{z}_2) \sin t \\ v_2(z,t) = -v_1(z,t) k_2 \dfrac{z_2}{\rho_{p,q}^2(\bar{z}_2)} - |v_1(z,t)| k_3 \dfrac{z_3}{\rho_{p,q}(\bar{z}_2)} \end{cases} \tag{49.34}$$

renders the origin of the three-dimensional chained system globally asymptotically stable, with a uniform exponential rate of convergence [49.7].

The parenthood of the controls (49.32) and (49.34) is noticeable. One can also verify that the control (49.34) is well defined (by continuity) at $\bar{z}_2 = 0$. More precisely, the ratios

$$\frac{z_2}{\rho_{p,q}^2(\bar{z}_2)} \quad \text{and} \quad \frac{z_3}{\rho_{p,q}(\bar{z}_2)} ,$$

which are obviously well defined when $\bar{z}_2 \neq 0$, tend to zero when $\bar{z}_2$ tends to zero. This guarantees the continuity of the control law.

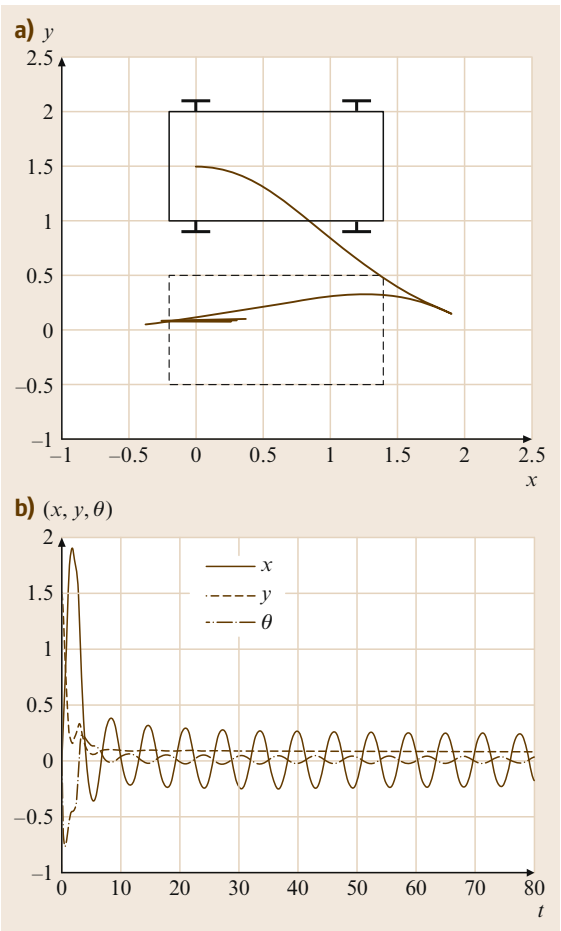


Fig.49.10a,b Asymptotic stabilization with a Lipschitz-continuous controller. **(a)** Cartesian motion; **(b)** error coordinates versus time

The property of exponential convergence pointed out in the above result calls for some remarks. Indeed, this property does not exactly correspond to the classical exponential convergence property associated with stable linear systems. In this latter case, exponential convergence means that for some constant K, γ, and along any solution to the controlled system, one has

$$\|z(t)\| \leq K \|z(t_0)\| e^{-\gamma(t-t_0)}$$

This corresponds to the common notion of *exponential stability*. In the present case, this inequality becomes

$$\rho[z(t)] \leq K \rho[z(t_0)] e^{-\gamma(t-t_0)}$$

for some function ρ, defined for example by

$$\rho(z) = |z_1| + \rho_{p,q}(z_2, z_3) ,$$

with $\rho_{p,q}$ as specified in Proposition 49.4. Although the function ρ shares common features with the Euclidean norm of the state vector (it is definite positive and it tends to infinity when $\|z\|$ tends to infinity), it is not equivalent to this norm. Of course, this does not change the fact that each component z_i of z converges to zero exponentially. However, the transient behavior is different because one only has

$$|z_i(t)| \leq K \|z(t_0)\|^{\alpha} e^{-\gamma(t-t_0)} ,$$

with $\alpha < 1$, instead of

$$|z_i(t)| \leq K \|z(t_0)\| e^{-\gamma(t-t_0)} .$$

In the case of the four-dimensional chained system, one can establish the following result, which is similar to Proposition 49.4.

Proposition 49.5

Let $\alpha, k_1, k_2, k_3, k_4 > 0$ be chosen such that the polynomial $p(s) = s^3 + k_4 s^2 + k_3 s + k_2$ is Hurwitz stable. For any integers $p, q \in \mathbb{N}^*$, let $\rho_{p,q}$ denote the function defined on $\mathbb{R}^3$ by

$$\rho_{p,q}(\bar{z}_2) = \left(|z_2|^{\frac{p}{q+2}} + |z_3|^{\frac{p}{q+1}} + |z_4|^{\frac{p}{q}} \right)^{\frac{1}{p}}$$

with $\bar{z}_2 = (z_2, z_3, z_4) \in \mathbb{R}^3$. Then, there exists $q_0 > 1$ such that, for any $q \geq q_0$ and $p > q + 2$, the continuous

Avidbots Ex1016
Page 1275 of 2154

state feedback

$$\begin{cases} v_1(z,t) = -k_1(z_1 \sin t - |z_1|) \sin t \\ +\alpha \rho_{p,q}(\bar{z}_2) \sin t \,, \\ v_2(z,t) = -|v_1(z,t)|k_2 \dfrac{z_2}{\rho^3_{p,q}(\bar{z}_2)} - v_1(z,t)k_3 \dfrac{z_3}{\rho^2_{p,q}(\bar{z}_2)} \\ -|v_1(z,t)|k_4 \dfrac{z_4}{\rho_{p,q}(\bar{z}_2)} \,, \end{cases} \tag{49.35}$$

renders the origin of the 4-D chained system globally asymptotically stable, with a uniform exponential rate of convergence [49.7].

The performance of the control law (49.35) is illustrated by the simulation results shown in Fig. 49.11. The control parameters have been chosen as follows: $\alpha = 0.6$, $k_{1,2,3,4} = (1.6, 10, 18, 17)$, $q = 2$, $p = 5$. The comparison with the simulation results of Fig. 49.10 shows a clear gain in performance.

49.4.5 Limitations Inherent to the Control of Nonholonomic Systems

Let us first mention some problems associated with the nonlinear time-varying feedbacks just presented. An ever important issue, when studying feedback control, is robustness. Indeed, if it were not for the sake of robustness, feedback control would lose much of its value and interest with respect to open-loop control solutions. There are various robustness issues. One of them concerns the sensitivity to modeling errors. For instance, in the case of a unicycle-type robot whose kinematic equations are in the form $\dot{\boldsymbol{x}} = u_1 b_1(\boldsymbol{x}) + u_2 b_2(\boldsymbol{x})$, one would like to know whether a feedback law which stabilizes an equilibrium of this system also stabilizes this equilibrium for the *neighbor* system $\dot{\boldsymbol{x}} = u_1[b_1(\boldsymbol{x}) + \varepsilon g_1(\boldsymbol{x})] + u_2[b_2(\boldsymbol{x}) + \varepsilon g_2(\boldsymbol{x})]$, with g_1 and g_2 denoting continuous applications, and ε a parameter which quantifies the modeling error. This type of error can account, for example, for a small uncertainty concerning the orientation of the actuated wheels axle with respect to the chassis, which results in a bias in the measurement of this orientation. One can show that time-varying control laws like (49.34) are not robust with respect to this type of error in the sense that, for certain functions g_1 and g_2, and for ε arbitrarily small, the system's solutions end up oscillating in the neighborhood of the origin, instead of converging to the origin. In other words, both the properties of stability of the origin and of convergence to this point can be jeopardized by arbitrarily small modeling errors, even in the absence of measurement noise. In view of these problems, one is brought to question the existence of fast (exponential) stabilizers endowed with robustness properties similar to those of stabilizing linear feedbacks for linear systems. The answer is that, to our knowledge, no such control solution (either continuous or discontinuous) has ever been found. More than likely such a solution does not exist for nonholonomic systems. Robustness of the stability property against modeling errors, and control discretization and delays, has been proved in some cases, but this could only be achieved with Lipschitz-continuous feedbacks which, as we have seen, yield slow convergence. The classical compromise between robustness and performance thus seems much more acute than in the case of stabilizable linear systems (or nonlinear systems whose linear approximation is stabilizable).

A second issue is the proven nonexistence of a *universal* feedback controller capable of stabilizing any feasible reference state trajectory asymptotically [49.8].

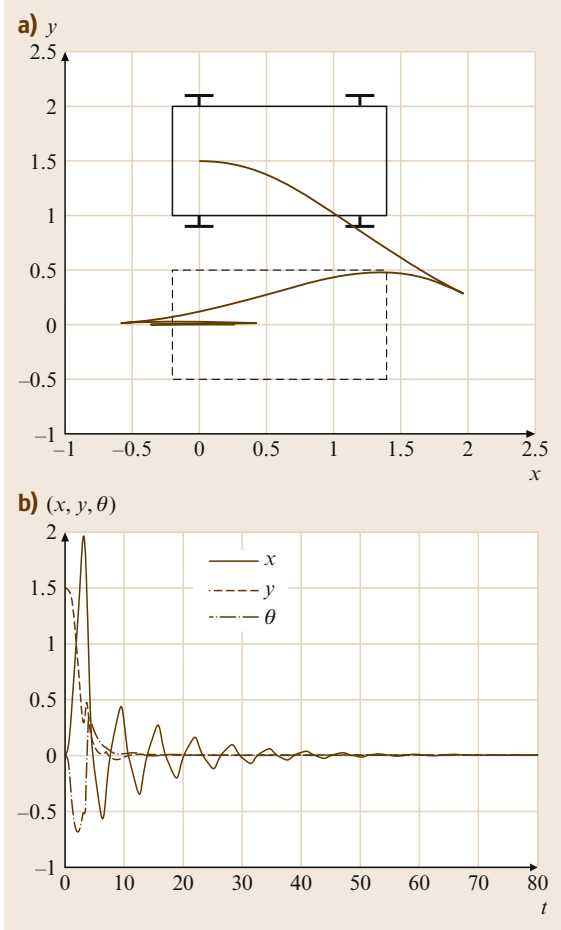


Fig.49.11a,b Asymptotic stabilization with a continuous (non-Lipschitz) time-varying feedback. **(a)** Cartesian motion; **(b)** error coordinates versus time

This is another notable difference with the linear case. Indeed, given a controllable linear system $\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u}$, the feedback controller $\boldsymbol{u} = \boldsymbol{u}_\mathrm{r} + \mathbf{K}(\boldsymbol{x} - x_\mathrm{r})$, with $\mathbf{K}$ a gain matrix such that $\mathbf{A} + \mathbf{BK}$ is Hurwitz stable, exponentially stabilizes any feasible reference trajectory $\boldsymbol{x}_\mathrm{r}$ (solution to the system) associated with the control input $\boldsymbol{u}_\mathrm{r}$. The nonexistence of such a controller, in the case of nonholonomic mobile robots, is related to the conditions upon the longitudinal velocity stated in previous propositions concerning trajectory stabilization. This basically indicates that such conditions cannot be removed entirely: whatever the chosen feedback controller, there always exists a feasible reference trajectory that this feedback cannot asymptotically stabilize. Note that this limitation persists when considering nonstandard feedbacks (such as, e.g., time-varying periodic feedbacks capable of asymptotically stabilizing reference trajectories which are reduced to a single point). Moreover, it has clear practical consequences because there are applications (automatic tracking of a human-driven car, for instance) for which the reference trajectory, and thus its properties, are not known in advance (is the leading car going to keep moving or stop?) so that one cannot easily decide on which controller to use. Switching between various controllers is a possible strategy, which has been studied by some authors and may give satisfactory results in many situations. However, since implementing a predefined switching strategy between two controllers sums up to designing a third controller, this does not solve the core of the problem nor grant any certitude of success.

A third issue, which is not specific to nonholonomic systems, but has seldom been addressed in the nonlinear control literature, concerns the problem of tracking *nonfeasible* trajectories (i. e., trajectories which are not solutions to the system's equations). Since exact tracking is not possible, by the definition of a nonfeasible trajectory, the control objective is then to ensure that the tracking errors shrink to, and ever after never exceed, certain nonzero thresholds. The fact that these thresholds can theoretically be arbitrarily small in the case of nonholonomic systems, if the amplitude of the velocity control inputs is not limited, makes this problem particularly relevant for these systems. This can be termed as a *practical* stabilization objective which, although slightly less ambitious than the objective of asymptotic stabilization considered in previous sections, opens up both the control design problem and the range of applications significantly. For instance, it allows one to address the problem of tracking an omnidirectional vehicle with a unicycle-type, or a car-like, vehicle. In the context of planning a trajectory with obstacle avoidance, transforming a nonfeasible trajectory into a feasible approximation for a certain mobile robot can be performed by applying a practical stabilizer to a model of this robot and by numerical integration of the system's closed-loop equations. Also, if one reformulates the former question about the existence of a *universal* stabilizer, with the objective of asymptotic stabilization now replaced by the one of practical stabilization, then the answer becomes positive: such a stabilizer exists and, moreover, the reference trajectories do not even have to be feasible.

49.4.6 Practical Stabilization of Arbitrary Trajectories

A possible approach for the design of practical stabilizers in the case of controllable driftless systems is described in [49.9]. Some of its basic principles, here adapted to the specific examples of unicycle-type and car-like mobile robots, are recalled next.

Let us introduce some matrix notation that will be used in this section.

$$\mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

$$\bar{\mathbf{R}}(\theta) = \begin{pmatrix} R(\theta) & 0 \\ 0 & 1 \end{pmatrix}.$$

Unicycle Case

With the above notation, the kinematic model (49.4) can be written as

$$\dot{\boldsymbol{g}} = \bar{\mathbf{R}}(\theta)\mathbf{C}\boldsymbol{u}\,, \tag{49.36}$$

with $\boldsymbol{g} = (\boldsymbol{x}, \boldsymbol{y}, \theta)'$ and

$$\mathbf{C} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Let us now consider a smooth function

$$f : \alpha \longmapsto f(\alpha) = \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \\ f_\theta(\alpha) \end{pmatrix},$$

with $\alpha \in S^1 = \mathbb{R}/2\pi\mathbb{Z}$ (i. e., α is an angle variable), and define

$$\bar{\boldsymbol{g}} := \begin{pmatrix} \bar{\boldsymbol{x}} \\ \bar{\boldsymbol{y}} \\ \bar{\theta} \end{pmatrix} := \boldsymbol{g} - \bar{\mathbf{R}}\,(\theta - f_\theta(\alpha))f(\alpha)$$

$$= \begin{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \mathbf{R}(\theta - f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ \theta - f_\theta(\alpha) \end{pmatrix}. \tag{49.37}$$

Note that $\bar{g}$ can be viewed as the situation of a frame $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ the origin of which has components

$$-\mathbf{R}(-f_\theta(\alpha))\begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix}$$

in the frame $\mathcal{F}_{\mathrm{m}}$. In term of differential geometry, $\bar{g}$ is the product of g by the inverse $f(\alpha)^{-1}$ of $f(\alpha)$, in the sense of the Lie group operations in $SE(2)$. Hence, $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ is all the closer to $\mathcal{F}_{\mathrm{m}}$ as the components of $f(\alpha)$ are small. For any smooth time function $t \longmapsto \alpha(t)$, and along any solution to system (49.36), the time derivative of $\bar{g}$ is given by

$$\dot{\bar{g}} = \bar{\mathbf{R}}(\bar{\theta})\bar{\boldsymbol{u}} \,, \tag{49.38}$$

with

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha)\left(\bar{\mathbf{R}}(f_\theta(\alpha)) - \frac{\partial f}{\partial \alpha}(\alpha)\right)\begin{pmatrix} \mathbf{C}\boldsymbol{u} \\ \dot{\alpha} \end{pmatrix}, \tag{49.39}$$

and

$$\mathbf{A}(\alpha) = \begin{pmatrix} I_2 & -S\begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ 0 & 1 \end{pmatrix}. \tag{49.40}$$

From (49.38) and (49.39), one can view $\dot{\alpha}$ as a complementary control input that can be used to monitor the motion of the frame $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$. More precisely, $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ can be viewed as an omnidirectional frame provided that $\bar{\boldsymbol{u}}$ can be rendered equal to any vector of $\mathbb{R}^3$, i. e., provided that the mapping $(\boldsymbol{u}, \dot{\alpha}) \longmapsto \bar{\boldsymbol{u}}$ is onto. Let us determine when this condition is satisfied. Equation (49.39) can also be written as

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha)\mathbf{H}(\alpha)\begin{pmatrix} \boldsymbol{u} \\ \dot{\alpha} \end{pmatrix}, \tag{49.41}$$

with

$$\mathbf{H}(\alpha) = \begin{pmatrix} \cos f_\theta(\alpha) & 0 & -\frac{\partial f_x}{\partial \alpha}(\alpha) \\ \sin f_\theta(\alpha) & 0 & -\frac{\partial f_y}{\partial \alpha}(\alpha) \\ 0 & 1 & -\frac{\partial f_\theta}{\partial \alpha}(\alpha) \end{pmatrix}. \tag{49.42}$$

Since $\mathbf{A}(\alpha)$ is invertible, $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ is omnidirectional if and only if the matrix $\mathbf{H}(\alpha)$ is also invertible. A function f which satisfies this property for *any* $\alpha \in S^1$ is called a *transverse function* [49.10]. The issue of the existence of such functions has been treated in the much more general context of the transverse function approach [49.9, 10]. In the present case, a family of transverse functions is given by

$$f(\alpha) = \begin{pmatrix} \varepsilon \sin\alpha \\ \varepsilon^2 \eta \dfrac{\sin 2\alpha}{4} \\ \arctan(\varepsilon\eta\cos\alpha) \end{pmatrix} \quad \text{with} \quad \varepsilon, \eta > 0 \,. \tag{49.43}$$

Indeed, with this function one can verify that, for any $\alpha \in S^1$,

$$\det\mathbf{H}(\alpha) = -\frac{\varepsilon^2\eta}{2}\cos(\arctan(\varepsilon\eta\cos\alpha)) < 0 \,.$$

Note that the components of f uniformly tend to zero as ε tends to zero, so that the associated omnidirectional frame $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ can be made arbitrarily close to $\mathcal{F}_{\mathrm{m}}$ by choosing ε small (but different from zero).

Now, let

$$t \longmapsto \boldsymbol{g}_{\mathrm{r}}(t) = [\boldsymbol{x}_{\mathrm{r}}(t), \boldsymbol{y}_{\mathrm{r}}(t), \theta_{\mathrm{r}}(t)]^{\mathrm{T}}$$

denote a smooth, but otherwise *arbitrary*, reference trajectory. It is not difficult to derive from (49.38) a feedback law $\bar{\boldsymbol{u}}$ which asymptotically stabilizes $\bar{\boldsymbol{g}}$ at $\boldsymbol{g}_{\mathrm{r}}$. A possible choice is given by

$$\bar{\boldsymbol{u}} = \bar{\mathbf{R}}(-\bar{\theta})[\dot{\boldsymbol{g}}_{\mathrm{r}} - k(\bar{\boldsymbol{g}} - \boldsymbol{g}_{\mathrm{r}})] \,, \tag{49.44}$$

which implies that

$$(\dot{\bar{\boldsymbol{g}}} - \dot{\boldsymbol{g}}_{\mathrm{r}}) = -k(\bar{\boldsymbol{g}} - \boldsymbol{g}_{\mathrm{r}})$$

and therefore that $\bar{g} - g_{\mathrm{r}} = 0$ is an exponentially stable equilibrium of the above equation for any $k > 0$. Then, it follows from (49.37) that

$$\lim_{t \to +\infty} \{g(t) - g_{\mathrm{r}}(t) - \bar{\mathbf{R}}[\theta_{\mathrm{r}}(t)]f[\alpha(t)]\} = 0 \,. \tag{49.45}$$

The norm of the tracking error $\|\boldsymbol{g} - \boldsymbol{g}_{\mathrm{r}}\|$ is thus ultimately bounded by the norm of $f(\alpha)$ which, in view of (49.43), can be made arbitrarily small via the choice of ε. It is in this sense that practical stabilization is achieved. The control $\boldsymbol{u}$ for the unicycle-like robot is then calculated by inverting the relation (49.41) and using the expression (49.44) of $\bar{\boldsymbol{u}}$.

While it can be tempting to use very small values of ε for the transverse function f in order to obtain a good tracking precision, one must be aware of the limits of this strategy. Indeed, when ε tends to

zero, the matrix $\mathbf{H}(\alpha)$ defined by (49.42) becomes ill-conditioned, and its determinant tends to zero. This implies, by (49.41), that the robot's velocities u_1 and u_2 may become very large. In particular, when the reference trajectory $\boldsymbol{g}_\mathrm{r}$ is not feasible, many manoeuvres are likely to occur. Note that this difficulty is intrinsic to the robot's nonholonomy and that it cannot be circumvented (think about the problem of parking a car in a very narrow parking place). For this reason, trying to impose very accurate tracking of nonfeasible trajectories is not necessarily a good option in practice. On the other hand, when the trajectory is feasible, manoeuvres are not needed to achieve accurate tracking, so that smaller values of ε can be used in this case. This clearly leads to a dilemma when the reference trajectory is not known in advance and its properties in term of feasibility can vary with time. A control strategy which addresses this issue, based on the use of transverse functions whose magnitude can be adapted online, is proposed in [49.11]. Experimental validations of the present approach on a unicycle-like robot can also be found in [49.12].

Car Case

The control approach presented above can be extended to car-like vehicles (and also to the trailer case). Again, the idea is to associate with the robot's frame $\mathcal{F}_\mathrm{m}$ an omnidirectional *companion* frame $\bar{\mathcal{F}}_\mathrm{m}(\alpha)$ which can be maintained arbitrarily close to $\mathcal{F}_\mathrm{m}$ via the choice of some design parameters. Let us show how this can be done for a car-like vehicle. To simplify the forthcoming equations, let us rewrite system (49.5) as

$$\begin{cases} \dot{x} = u_1 \cos\theta \,, \\ \dot{y} = u_1 \sin\theta \,, \\ \dot{\theta} = u_1 \xi \,, \\ \dot{\xi} = u_\xi \,, \end{cases}$$

with $\xi = (\tan\phi)/L$ and $u_\xi = u_2(1+\tan^2\phi)/L$. This system can also be written as (compare with (49.36))

$$\begin{cases} \dot{\boldsymbol{g}} = \bar{\mathbf{R}}(\theta)\mathbf{C}(\xi)u_1 \,, \\ \dot{\xi} = \boldsymbol{u}_\xi \,, \end{cases} \tag{49.46}$$

with $\boldsymbol{g} = (\boldsymbol{x}, \boldsymbol{y}, \theta)^\mathrm{T}$ and $\mathbf{C}(\xi) = (1, 0, \xi)^\mathrm{T}$. Let us now consider a smooth function

$$f : \alpha \longmapsto f(\alpha) = \begin{pmatrix} f_g(\alpha) \\ f_\xi(\alpha) \end{pmatrix} = \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \\ f_\theta(\alpha) \\ f_\xi(\alpha) \end{pmatrix},$$

with $\alpha \in S^1 \times S^1$ (i. e., $\alpha = (\alpha_1, \alpha_2)$), and define (compare with (49.37))

$$\bar{\boldsymbol{g}} := \begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{pmatrix} := \boldsymbol{g} - \bar{\mathbf{R}}(\theta - f_\theta(\alpha)) f_g(\alpha)$$

$$= \begin{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \mathbf{R}(\theta - f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ \theta - f_\theta(\alpha) \end{pmatrix}, \tag{49.47}$$

which, as in the unicycle case, can be viewed as the situation of some *companion* frame $\bar{\mathcal{F}}_\mathrm{m}(\alpha)$. By differentiating $\bar{\boldsymbol{g}}$ along any smooth time function $t \longmapsto \alpha(t)$ and any solution to system (49.46), one can verify that (49.38) is still satisfied, except that $\bar{\boldsymbol{u}}$ is now given by

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha) \left(\bar{\mathbf{R}}[f_\theta(\alpha)] - \frac{\partial f_g}{\partial \alpha_1}(\alpha) - \frac{\partial f_g}{\partial \alpha_2}(\alpha) \right) \times \begin{pmatrix} \mathbf{C}(\xi)u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} \tag{49.48}$$

rather than by (49.39) (with $\mathbf{A}(\alpha)$ still defined by (49.40)). Using the fact that

$$\mathbf{C}(\xi)u_1 = \mathbf{C}(f_\xi(\alpha))u_1 + \{\mathbf{C}(\xi) - \mathbf{C}[f_\xi(\alpha)]\}u_1 = \begin{pmatrix} 1 \\ 0 \\ f_\xi(\alpha) \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ \xi - f_\xi(\alpha) \end{pmatrix} u_1 \,,$$

(49.48) can also be written as

$$\bar{\boldsymbol{u}} = A(\alpha)\mathbf{H}(\alpha) \begin{pmatrix} u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} + A(\alpha) \begin{pmatrix} 0 \\ 0 \\ u_1[\xi - f_\xi(\alpha)] \end{pmatrix}, \tag{49.49}$$

with

$$\mathbf{H}(\alpha) = \begin{pmatrix} \cos f_\theta(\alpha) & -\frac{\partial f_x}{\partial \alpha_1}(\alpha) & -\frac{\partial f_x}{\partial \alpha_2}(\alpha) \\ \sin f_\theta(\alpha) & -\frac{\partial f_y}{\partial \alpha_1}(\alpha) & -\frac{\partial f_y}{\partial \alpha_2}(\alpha) \\ f_\xi(\alpha) & -\frac{\partial f_\theta}{\partial \alpha_1}(\alpha) & -\frac{\partial f_\theta}{\partial \alpha_2}(\alpha) \end{pmatrix}. \tag{49.50}$$

By setting

$$\boldsymbol{u}_\xi = \dot{f}_\xi(\alpha) - k(\xi - f_\xi(\alpha) \,, \tag{49.51}$$

Avidbots Ex1016
Page 1279 of 2154

with $k > 0$, it follows from (49.46) that $\xi - f_\xi(\alpha)$ exponentially converges to zero. Hence, after some transient phase whose duration is commensurable with $1/k$, $\xi - f_\xi(\alpha) \approx 0$, and (49.49) reduces to

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha)\mathbf{H}(\alpha)\begin{pmatrix} u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix}. \tag{49.52}$$

Provided that the function f is such that $\mathbf{H}(\alpha)$ is always invertible, this latter relation means that the frame $\bar{\mathcal{F}}_\mathrm{m}(\alpha)$ associated with $\bar{\boldsymbol{g}}$ is omnidirectional. Any function f for which this property is satisfied is called a transverse function. Once it has been determined, one can proceed as in the unicycle case to asymptotically stabilize an arbitrary reference trajectory $\boldsymbol{g}_\mathrm{r}$ for $\bar{\boldsymbol{g}}$, for example, by defining $\bar{\boldsymbol{u}}$ as in (49.44). The control u_1 for the car is then obtained by inverting relation (49.52).

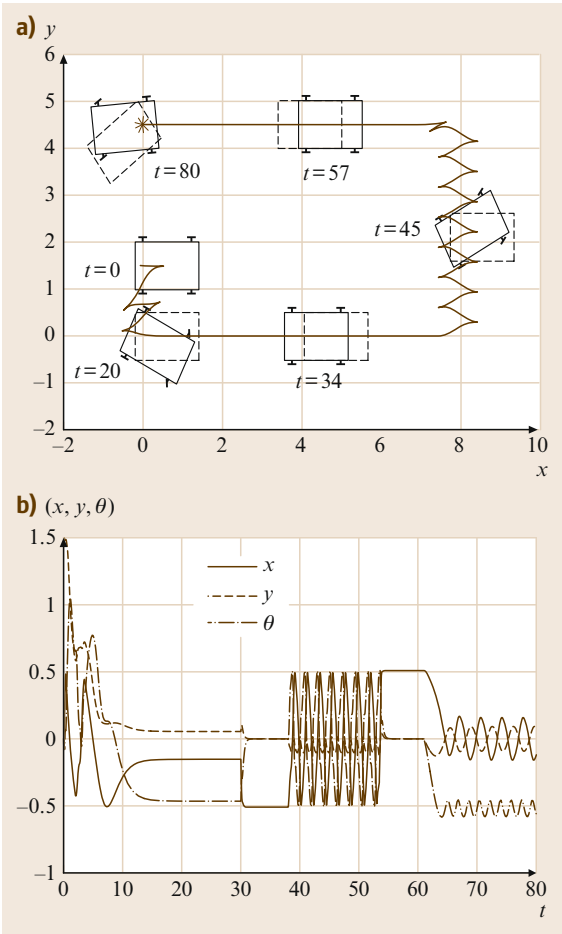


Fig.49.12a,b Practical stabilization of an arbitrary trajectory by the transverse function approach. **(a)** Cartesian motion; **(b)** error coordinates versus time

The following lemma specifies a family of transverse functions for the car case.

Lemma 49.1

For any $\varepsilon > 0$ and any η_1, η_2, η_3 such that $\eta_1, \eta_2, \eta_3 > 0$ and $6\eta_2\eta_3 > 8\eta_3 + \eta_1\eta_2$, the function f defined by

$$f(\alpha) = \begin{pmatrix} \bar{f}_1(\alpha) \\ \bar{f}_4(\alpha) \\ \arctan(\bar{f}_3(\alpha)) \\ \bar{f}_2(\alpha)\cos^3 f_3(\alpha) \end{pmatrix}$$

with

$$\bar{f} : S^1 \times S^1 \longrightarrow \mathbb{R}^4$$

given by

$$\bar{f}(\alpha) = \begin{pmatrix} \varepsilon(\sin\alpha_1 + \eta_2 \sin\alpha_2) \\ \varepsilon\eta_1 \cos\alpha_1 \\ \varepsilon^2\left(\dfrac{\eta_1 \sin 2\alpha_1}{4} - \eta_3 \cos\alpha_2\right) \\ \varepsilon^3\left(\eta_1 \dfrac{\sin^2\alpha_1 \cos\alpha_1}{6} - \dfrac{\eta_2\eta_3 \sin 2\alpha_2}{4} - \eta_3 \sin\alpha_1 \cos\alpha_2\right) \end{pmatrix},$$

satisfies the transversality condition $\det\mathbf{H}(\alpha) \neq 0\ \forall\alpha$, with $\mathbf{H}(\alpha)$ defined by (49.50) [49.13].

The simulation results reported in Fig. 49.12 illustrate the application of this control approach for a car-like robot. The reference trajectory is defined by the initial condition $g_\mathrm{r}(0) = 0$ and its time derivative

$$\dot{g}_\mathrm{r}(t) = \begin{cases} (0,0,0)^\mathrm{T} & \text{if } t \in [0,30] \\ (1,0,0)^\mathrm{T} & \text{if } t \in [30,38] \\ (0,0.3,0)^\mathrm{T} & \text{if } t \in [38,53] \\ (-1,0,0)^\mathrm{T} & \text{if } t \in [53,61] \\ (0,0,0.2)^\mathrm{T} & \text{if } t \in [61,80] \end{cases}.$$

This corresponds to a fixed situation when $t \in [0,30]$, three sequences of pure translational motion when $t \in [30,61]$, and a pure rotational motion when $t \in [61,80]$. Let us remark that this trajectory is not feasible for the car-like robot when $t \in [38,53]$, since it corresponds to a lateral translation in the direction of the unit vector $\boldsymbol{j}_\mathrm{r}$ of the frame $\mathcal{F}_\mathrm{r}$ associated with g_r, nor when $t \in [61,80]$, since a rear-drive car cannot perform pure rotational motion. The initial configuration of the car-like robot, at $t = 0$, is $\boldsymbol{g}(0) = (0, 1.5, 0)$, and the initial steering wheel angle is $\phi(0) = 0$.

In Fig. 49.12a, the robot is drawn with a solid line at several time instants, whereas the chassis of the reference vehicle is shown as a dashed line at the same time instants. The figure also shows the trajectory of the point located at the mid-distance of the robot's rear wheels. Figure 49.12b shows the time evolution of the tracking error expressed in the reference frame (i. e., (x_e, y_e) as defined by (49.16), and $\theta_e = \theta - \theta_r$). It follows from (49.45) that, after the transient phase associated with the exponential convergence of $\bar{g}$ to zero, the ultimate bound for $|x_e|$, $|y_e|$, and $|\theta_e|$ is upper-bounded by the maximum amplitude of the functions f_x, f_y, and f_θ, respectively. For this simulation, the control parameters of the transverse function f of Lemma 49.1 have been chosen as follows: $\varepsilon = 0.17$, $\eta_{1,2,3} = (12, 2, 20)$. With these values, one can verify that $|f_x|$, $|f_y|$, and $|f_\theta|$ are bounded by 0.51, 0.11, and 0.6, respectively. This is consistent with the time evolution of the tracking error observed in the figure. As pointed out for the unicycle case, tracking errors could be diminished by decreasing the value of ε, but this would involve larger values of the control inputs and also more frequent manoeuvres, especially on the time intervals [38, 53] and [61, 80] when the reference trajectory is not feasible. Additional simulation and experimental results on this approach are illustrated by VIDEO 182 and VIDEO 243.

49.5 Path Following in the Case of Nonideal Wheel–Ground Contact

The kinematic models used in the previous sections are derived under the classical rolling-without-sliding assumption for the vehicle's wheels. This assumption is satisfied with a good degree of accuracy for many applications both indoor and outdoor (e.g., on-road). In some cases, however, sliding can be significant. This happens for example when a vehicle operates on natural terrain with poor grip conditions (grass, earth) and when the vehicle's speed is significant and/or the terrain is not perfectly horizontal. In these cases, the control laws presented in the previous sections may not give full satisfaction. We show in this section that these control laws can still be used successfully, provided that sliding is taken into account at the modelling level and estimated online via a dedicated observer. For simplicity, only the path following problem is addressed, but the techniques here presented can be extended to other control problems.

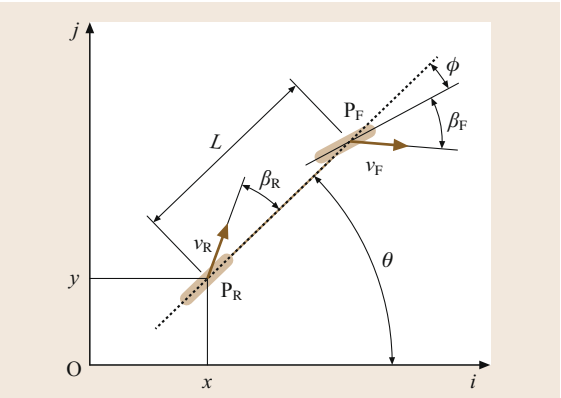


Fig. 49.13 Car-like vehicle in the presence of sliding

49.5.1 Extended Control Models in the Presence of Sliding

Extended Kinematic Model

Consider the two-wheels schematic representation of a car-like vehicle on Fig. 49.13. The angles β_R and β_F are introduced in order to represent the sliding of the rear and front wheels respectively. More precisely, denoting the centres of the rear and front wheels as P_R and P_F respectively, β_R is the angle between the vector $\boldsymbol{P_R P_F}$ and the velocity vector $\boldsymbol{v}_R$ of $\boldsymbol{P}_R$, whereas β_F represents the angle between the steering direction and the velocity vector $\boldsymbol{v}_F$ of $\boldsymbol{P}_F$. The kinematic modeling of Sect. 49.2 is easily extended to the present case [49.14], resulting in the following model

$$\begin{cases} \dot{x} = u_1 \cos(\theta + \beta_R) \\ \dot{y} = u_1 \sin(\theta + \beta_R) \\ \dot{\theta} = u_1 \cos(\beta_R) \dfrac{\tan(\phi + \beta_F) - \tan(\beta_R)}{L} \\ \dot{\phi} = u_2 \end{cases} , \tag{49.53}$$

with u_1 the (signed) intensity of the vector $\boldsymbol{v}_R$. Note that these equations reduce to (49.5) when $\beta_F = \beta_R = 0$.

Kinematic Model in a Frénet Frame

In the context of path following, a kinematic model with reference to a Frénet frame moving along the desired path is obtained via a straightforward adaptation of the pure-rolling case discussed in Sect. 49.2.3. Defining the distance between the vehicle and the desired path C as the distance d between the point P_R and this path (Fig. 49.5), the equations of this model are readily de-

duced from (49.53) (compare with (49.13))

$$\begin{cases} \dot{s} = u_1 \dfrac{\cos(\theta_e + \beta_R)}{1 - d\,c(s)} \\ \dot{d} = u_1 \sin(\theta_e + \beta_R) \\ \dot{\theta}_e = u_1 \left[\cos(\beta_R)\lambda_1 - \lambda_2\right] \\ \dot{\phi} = u_2 \end{cases}, \tag{49.54}$$

with θ_e the angle between the vehicle's body axis $P_R P_F$ and the tangent to the desired path evaluated at the projection of the point P_R on this path, $c(s)$ the path curvature at the projected point, $\lambda_1 = (\tan(\phi + \beta_F) - \tan(\beta_R))/L$ and $\lambda_2 = (c(s)\cos(\theta_e + \beta_R))/(1 - d\,c(s))$.

Dynamic Model of Sliding Angles

The kinematic model (49.53) can be used for control design once it is completed by a model of the dynamics of the sliding angles β_R and β_F. Such a model can be obtained from Newton's law and a model of tire/ground interactions. A few notation (Fig. 49.14 for details) and assumptions are introduced for this purpose:

- The vehicle's mass is denoted as m and its moment of inertia with respect to the (body-fixed) vertical axis is denoted as I_z. The vehicle's center of mass G is located on the segment joining P_R to P_F, at a distance L_R from P_R and L_F from P_F.
- The longitudinal dynamics is neglected. More precisely, it is assumed that the traction force applied to the vehicle, in relation to the monitoring of longitudinal tire/ground contact forces, is controlled independently of the vehicle's lateral dynamics and that, as a result of this control, the longitudinal velocity of the vehicle expressed in body frame, i. e., $u_1 \cos\beta_r$, varies slowly so that its time-derivative can be neglected in the calculus of the lateral dynamics.
- Lateral tire/ground contact forces, that are orthogonal to the wheels planes, are denoted as $\boldsymbol{F}_R$ and $\boldsymbol{F}_F$ respectively.
- The lateral component of the gravity force is denoted as $\boldsymbol{F}_G$. This force applies at G and its magnitude is $mg\sin\alpha$ with α denoting the terrain slope angle in the lateral direction.

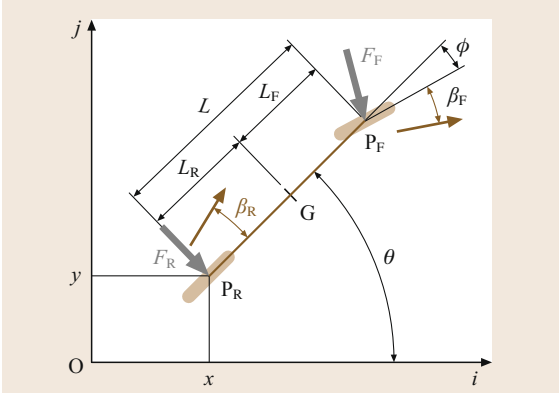


Fig. 49.14 Lateral forces applying to a car-like vehicle

Application of Newton's law yields [49.15], after projection in the direction orthogonal to $\boldsymbol{v}_R$,

$$\dot{\beta}_R = \frac{1}{mu_1}\Big[(F_R + mg\sin\alpha)\cos\beta_R + F_F\cos(\beta_R - \phi) - mL_R\ddot{\theta}\cos\beta_R\Big] - \dot{\theta}$$

with F_F and F_R the (signed) intensity of the lateral forces $\boldsymbol{F}_F$ and $\boldsymbol{F}_R$ and

$$\ddot{\theta} = \frac{1}{I_z}(L_F F_F\cos\phi - L_R F_R) \tag{49.55}$$

From (49.53)

$$\tan(\phi + \beta_F) = \tan\beta_R - \frac{L\dot{\theta}}{u_1\cos\beta_R}\,.$$

Differentiating this equation with respect to time and using the assumption that $u_1\cos\beta_R$ is constant yields

$$\dot{\beta}_F = \cos^2(\phi + \beta_F)\left(\frac{\dot{\beta}_R}{\cos^2\beta_R} - L\frac{\ddot{\theta}}{u_1\cos\beta_R}\right) - \dot{\phi}\,.$$

The dynamics of the sliding angles is thus given by

$$\begin{cases} \dot{\beta}_R = \dfrac{1}{mu_1}\Big[(F_R + mg\sin\alpha)\cos\beta_R + F_F\cos(\beta_R - \phi) - mL_R\ddot{\theta}\cos\beta_R\Big] - \dot{\theta}\,, \\ \dot{\beta}_F = \cos^2(\phi + \beta_F)\left(\dfrac{\dot{\beta}_R}{\cos^2\beta_R} - L\dfrac{\ddot{\theta}}{u_1\cos\beta_R}\right) - \dot{\phi}\,. \end{cases} \tag{49.56}$$

Replacing the vehicle's angular acceleration $\ddot{\theta}$ by its expression (49.55), and $\dot{\theta}, \dot{\phi}$ by (49.53), one obtains expressions of the sliding angles dynamics in terms of ϕ, α, u_1, u_2, F_R, and F_F.

Tire/Ground Interaction Models for Lateral Forces Computation

In view of (49.56), knowledge of the lateral ground forces $\boldsymbol{F}_R$ and $\boldsymbol{F}_F$ is needed to calculate the evolution of the sliding angles and the vehicle's motion. In

this respect, a model of tire/ground interactions is useful, in particular for simulation purposes. The Coulomb model, although very popular, does not allow one to describe the complexity of tire/ground interactions in a large operating domain. For this reason, other contact models have been proposed in the literature. For instance, the *Dahl* [49.16] and LuGre [49.17] models, eventhough they are not dedicated to tire/ground contact description, give a relationship between contact force and sliding velocity with a small number of parameters. They are used to describe a vehicle's dynamics in [49.18]. Of particular interest is the celebrated tire/ground contact model of Pacejka et al. [49.19], of which several versions depending on the application have been derived. A convenient model for mobile robot simulation and analysis is the so-called *magic formula* proposed in [49.20]. Alike other contact models, this formula expresses the lateral ground force in term of the sliding angle. It is defined as follows,

$$\begin{cases} F_* = D\sin\left[c\arctan(B[1-E])\beta_* + \dfrac{E}{B}\arctan(B\beta_*)\right] \\ D = a_1(F_z^*)^2 + a_2F_z^* \\ E = a_6(F_z^*)^2 + a_7F_z^* + a_8 \\ B = \dfrac{a_3\sin(a_4\arctan(a_5F_z^*))}{cD} \end{cases}, \tag{49.57}$$

with $* \in \{F, R\}$ used to denote either the front (F) or Rear (R) wheel, and F_z^* the corresponding tire load here defined as $\frac{L_*}{L}m$. The parameters a_i, $i \in [1, \ldots, 8]$ and c are representative of the grip conditions and tire properties (pressure, contact patches). Table 49.1 shows typical values of these parameters for a vehicle moving on wet grass at a velocity of 2 m/s (Sect. 49.5.4 further on). Fig. 49.15 depicts the corresponding relation between the front lateral force F_F and the front sliding angle β_F. This relationship is symmetric with respect to the origin, is quasi linear for small angles, has an extremum near $\pm -10°$, and involves a saturation for large sliding angles. The linear part of the function is often sufficient to account for slow to moderately fast motion in the case of good grip conditions, whereas its nonlinear part becomes important when the vehicle moves on natural ground or at high speed with possibly large sliding angles.

Table 49.1 Parameters used for dynamic modelling

Coefficients	Values
a_1	-25
a_2	500
a_3	1000
a_4	2
a_5	1
a_6	0
a_7	-0.35
a_8	5
c	1.6
$L = L_F + L_R$	$1.3 = 0.6 + 0.7$
m	380
I_z	300

Tire/ground contact models are necessary to describe the sliding angles' dynamics with some accuracy and are useful to evaluate the performance of control laws in simulation. The main drawback of these models is the large number of involved parameters and the difficulty to evaluate appropriate values in practical applications, since such values heavily depend on imprecisely known ground characteristics which, furthermore, may rapidly change along the vehicle's trajectory. For these reasons, previously evoked models may not be best suited for control design. In many cases, rather than trying to exploit a complete model involving unreliable and rapidly varying parameters it is preferable to use cruder models associated with an on-line estimation procedure. The control design proposed further on follows this latter avenue.

49.5.2 On-Line Estimation of Sliding Angles

As already mentionned, some knowledge of the sliding angles β_F and β_R is useful to precisely control the lateral distance d between the vehicle and the desired path. Since direct measurement of these angles via the use of dedicated sensors is quite difficult in practice, an alternative consists in designing an observer that produces on-line estimates of the sliding angles based on the measurement of the relative vehicle/path position and orientation. The underlying implicit (crude) model used

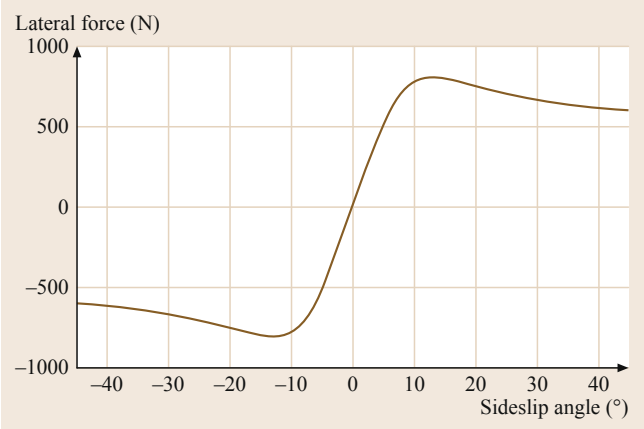


Fig. 49.15 Relationship between lateral force and sliding angle obtained using Pacejka Model

Avidbots Ex1016
Page 1283 of 2154

for the design of this observer is that the sliding angles do not change rapidly, as can be anticipated in common situations like driving along a road with a slowly varying curvature or following a straight line across a field with a constant slope. A solution of this type, exploiting also the vehicle's kinematic model (49.54), is proposed next.

Define $\xi = (d, \theta_e)^T$, $\beta = (\beta_F, \beta_R)^T$, and consider the function f defined by

$$f(\xi, \beta, \phi, c(s)) = \begin{pmatrix} \sin(\xi_2 + \beta_2) \\ \cos(\beta_2)\dfrac{\tan(\phi + \beta_1) - \tan(\beta_2)}{L} - \dfrac{c(s)\cos(\xi_2 + \beta_2)}{1 - \xi_1 c(s)} \end{pmatrix} .$$

From (49.54),

$$\dot{\xi} = u_1 f(\xi, \beta, \phi, c(s)) . \tag{49.58}$$

Define an observer of the form

$$\begin{cases} \dot{\hat{\xi}} = u_1 f(\xi, \hat{\beta}, \phi, c(s)) + \alpha_\xi , \\ \dot{\hat{\beta}} = \alpha_\beta . \end{cases} \tag{49.59}$$

The estimation error $(\tilde{\xi}, \tilde{\beta})^T = (\xi - \hat{\xi}, \beta - \hat{\beta})^T$ satisfies

$$\begin{cases} \dot{\tilde{\xi}} = u_1 [f(\xi, \beta, \phi, c(s)) - f(\xi, \hat{\beta}, \phi, c(s))] - \alpha_\xi , \\ \dot{\tilde{\beta}} = -\alpha_\beta + \dot{\beta} . \end{cases} \tag{49.60}$$

The objective is to define the terms α_ξ and α_β so as to ensure the asymptotic stability of the origin of the above estimation error system when β is constant, i. e., when $\dot{\beta} = 0$, and thus the asymptotic convergence of the estimate $\hat{\beta}$ to β in this case. Furthermore, if β varies slowly the estimation error should remain small.

Proposition 49.6

Let

$$\begin{cases} \alpha_\xi = |u_1| K_1 \tilde{\xi} \\ \alpha_\beta = K_2 u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \hat{\beta}, \phi, c(s)) \right]^T \tilde{\xi} \end{cases} \tag{49.61}$$

with K_1 a 2×2 positive definite matrix and K_2 a positive scalar. Assume that β is constant and that

i) The variables $u_1, \xi, \phi, c(s), 1 - dc(s)$ are bounded, differentiable, and their time-derivatives are bounded.

ii) There exists $\tau > 0$ such that

$$|\phi + \beta_1|, \ |\beta_2|, \ |\xi_2 + \beta_2| \leq \pi/2 - \tau .$$

iii) The function u_1 is persistently exciting in the sense that there exist two constants $T, \delta > 0$ such that

$$\forall t, \quad \int_t^{t+T} |u_1(s)| \, \mathrm{d}s \geq \delta .$$

Then, the origin of system (49.60) is locally exponentially stable.

Sketch of Proof: From (49.61), the estimation error dynamics (49.60) can be written as

$$\begin{cases} \dot{\tilde{\xi}} = u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \hat{\beta}, \phi, c(s))\tilde{\beta} + O^2(\tilde{\beta}) \right] - |u_1| K_1 \tilde{\xi} , \\ \dot{\tilde{\beta}} = -K_2 u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \hat{\beta}, \phi, c(s)) \right]^T \tilde{\xi} , \end{cases} \tag{49.62}$$

where $O^2(\tilde{\beta})$ denotes a second-order term in $\tilde{\beta}$ in a neighborhood of $\tilde{\beta} = 0$. Note that this term also depends on $\xi, \phi, c(s)$ and $1 - dc(s)$. The linearized system associated with system (49.62) is thus given by

$$\begin{cases} \dot{\tilde{\xi}} = u_1 \dfrac{\partial f}{\partial \beta}(\xi, \beta, \phi, c(s))\tilde{\beta} - |u_1| K_1 \tilde{\xi} , \\ \dot{\tilde{\beta}} = -K_2 u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \beta, \phi, c(s)) \right]^T \tilde{\xi} . \end{cases} \tag{49.63}$$

Consider the candidate Lyapunov function V defined by

$$V(\tilde{\xi}, \tilde{\beta}) = K_2 |\tilde{\xi}|^2 + |\tilde{\beta}|^2 .$$

One verifies that the time-derivative of V along the solutions of system (49.63) satisfies

$$\dot{V} = -|u_1| K_2 \tilde{\xi}^T K_1 \tilde{\xi} .$$

Since K_1 is positive definite and K_2 is positive, V is decreasing. This ensures the stability of the origin $(\tilde{\xi}, \tilde{\beta}) = 0$. Using assumptions i) and ii) and the fact that the matrix $\partial f / \partial \beta(\xi, \beta, \phi, c(s))$ is invertible one can show, that the origin of system (49.63) is asymptotically stable. Assumption iii) implies that the origin is in fact exponentially stable. Using assumptions i) and ii) again, one shows that the origin of the original (nonlinear) system (49.62) is locally exponentially stable. ■

Proposition 49.6 calls for a few remarks. Although convergence of the estimation errors is proved in the case of constant sliding angles only, the observer can still provide a good estimate of β if $\dot{\beta}$ is *small*. For fast variations of β, estimation based on a dynamic model and IMU measurements (inertial measurement

unit) will usually provide better results [49.21]. Satisfaction of assumptions i) and ii) is strongly related to the choice of the controller used to determine the steering wheel angular velocity u_2 in (49.54). In this respect, let us recall that the separation principle, which allows one to design independently a feedback controller and an observer for a linear system with a guarantee of stability of the coupled system, is not systematically satisfied for nonlinear systems.

49.5.3 Feedback Laws for Path Following

The kinematic model (49.54) can be transformed into a chained system. This is a direct extension of the results of Sect. 49.4.1. More precisely, the change of coordinates and control variables $(s, d, \theta_e, \phi, u_1, u_2) \longmapsto (z_1, z_2, z_3, z_4, v_1, v_2)$ defined by

$$
\begin{aligned}
&(z_1, z_2, z_3, z_4) = \\
&\Big\{ s,\ d,\ [1 - dc(s)] \tan(\theta_e + \beta_R), \\
&-c(s)[1 - dc(s)] \left[1 + 2\tan^2(\theta_e + \beta_R)\right] \\
&- d\frac{\partial c}{\partial s} \tan(\theta_e + \beta_R) \\
&+ [1 - dc(s)]^2 \frac{\tan(\phi + \beta_F) - \tan\beta_R}{L} \frac{1 + \tan^2(\theta_e + \beta_R)}{\cos(\theta_e + \beta_R)} \Big\}, \\
&(v_1, v_2) = (\dot{z}_1, \dot{z}_4)
\end{aligned}
$$

transforms the model (49.54) of a car-like vehicle into a 4-D chained system.

Based on this transformation, Sect. 49.4 provides a feedback control solution v_2 to the path following problem. Since the relation between v_2 and the original control variable u_2 involves the sliding angles β_F and β_R, u_2 can be computed from the outputs $\hat{\beta}_F$, $\hat{\beta}_R$ of the observer proposed in the previous section.

In practice, preexisting low-level control loops may use the steering angle ϕ itself, rather than its time-derivative, as a control input. In this case the expression for the *desired* steering angle can be obtained by identifying the expression of $\dot{\theta}_e$ in (49.54) with the one in (49.13), where u_2 is the feedback law deduced from the (chained-form) control law (49.27) of a unicycle-like robot. Further assuming that $c(s)$ is constant, one obtains the following expression

$$
\phi_{ref} = \arctan\left[\tan(\hat{\beta}_R) + \frac{L}{\cos(\hat{\beta}_R)} \left(\frac{c(s)\cos\tilde{\theta}_2}{\kappa} + \frac{A\cos^3\tilde{\theta}_2}{\kappa^2} \right) \right] - \hat{\beta}_F \tag{49.64}
$$

with

$$
\begin{cases}
\tilde{\theta}_2 = \theta_e + \hat{\beta}_R \\
\kappa = 1 - c(s)\, d \\
A = -k_2 d - k_3\, \kappa\, \mathrm{sign}(u_1) \tan\tilde{\theta}_2 + c(s)\kappa \tan^2\tilde{\theta}_2 \,.
\end{cases} \tag{49.65}
$$

49.5.4 Path Following Illustrations in Low Grip Conditions

Simulation Results

The simulation results presented next have been obtained by using the kinematic and dynamic models (49.54) and (49.56) respectively, with the set of parameters for the tire/ground interaction specified in Table 49.1. The reference path, depicted on Fig. 49.16 in black plain line, is composed of two straight lines connected by a circular path with a radius of 13 m. The simulated grip conditions generate sliding along the circular part of the path. The transitions between this circular part and the straight lines yield transient errors due to the curvature discontinuity. The vehicle's velocity is 3 m/s.

The steering velocity u_2 is computed according to the simple proportional feedback law

$$
u_2 = -k_\phi(\phi - \phi_{ref})
$$

with ϕ_{ref} defined by (49.64) and the control gain $k_\phi = 12$ (settling time of 250 ms). The other control gains in (49.64) are defined as $(k_2, k_3) = (0.09, 0.6)$, yielding a settling distance of 12m without overshoot. First,

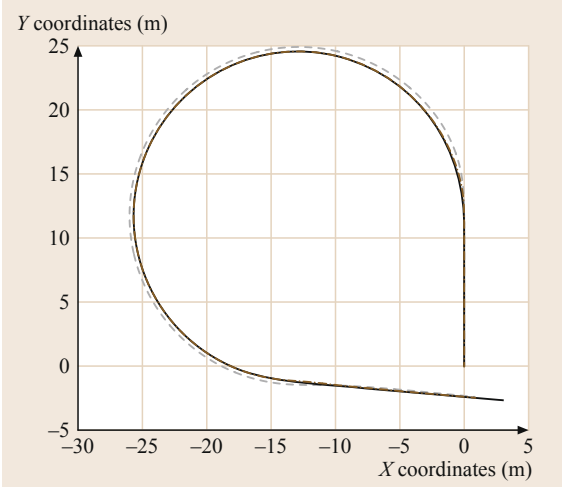


Fig. 49.16 Reference path and trajectories in Cartesian plane

Avidbots Ex1016
Page 1285 of 2154

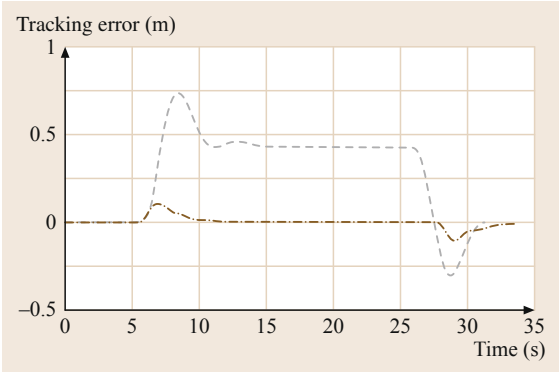


Fig. 49.17 Lateral tracking error (d)

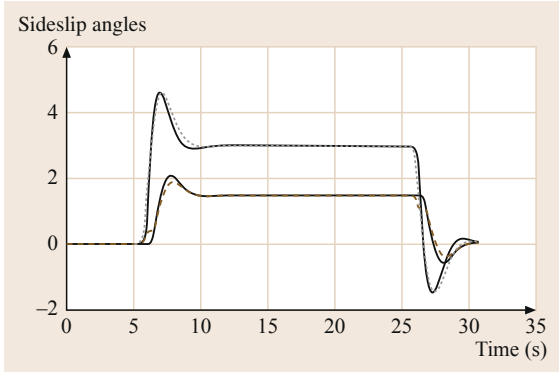


Fig. 49.18 Comparison of estimated and *actual* sliding angles

the control law is applied with $(\hat{\beta}_F, \hat{\beta}_R) = (0, 0)$ (i. e., without considering sliding). The results are shown in blue on Figs. 49.16 and 49.17. Then, the control law is applied with $\hat{\beta}_F$ and $\hat{\beta}_R$ obtained via the observer (49.59)–(49.61), using for K_1 a diagonal matrix with elements equal to 10 and 20 and the gain $K_2 = 10$. The results are shown in red. Fig. 49.17 clearly illustrates the reduction of lateral error resulting from the sliding angles estimation.

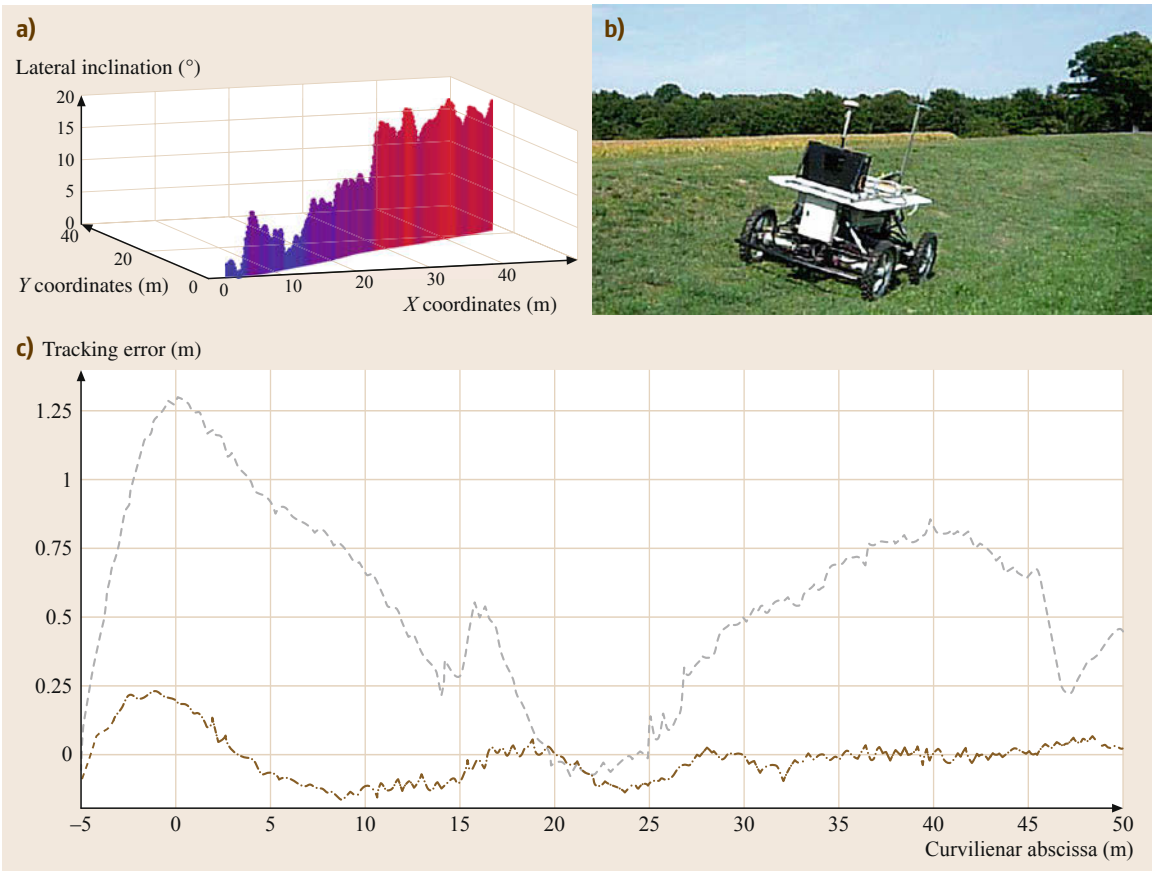


Fig.49.19a–c Path following with sliding effects for an autonomous off-road robot: **(a)** Reference trajectory in 3-D frame **(b)** Robot during autonomous tracking **(c)** Tracking errors (*dashed*: without sliding compensation, *dash-dotted*: with sliding estimation and compensation)

Performance of the proposed observer is illustrated on Fig. 49.18 with the *true* sliding angles (i. e., obtained from the simulated dynamics) in plain lines and the estimated angles in dashed lines. One can verify that the sliding angles are correctly estimated. In practice, using an observer allows one to also compensate for various unmodeled dynamics resulting, for instance, from imprecise knowledge of the vehicle's geometry (the distance L, for example) or from an unknown offset on the steering angle.

Experimental Results

Full scale experiments on natural terrain with the car-like vehicle of Fig. 49.19 are reported next and in VIDEO 435. This mobile robot is equipped with a real time kinematics global positioning system (RTK-GPS) producing a ± 2 cm accurate absolute position measurement. The GPS antenna is located straight above the control point P_R. The reference path, consisting of a straight line perpendicular to the slope direction, has been previously recorded during a manual driving. The top left of Fig. 49.19 shows the recorded coordinates in x-y plane, and the z axis represents the absolute value of the lateral slope angle α. This angle progressively reaches the value of 10° at the curvilinear abscissa 27 m, and it subsequently rapidly increases up to about 18°.

As for the simulation results reported previously, the feedback control (49.64) was first applied with

$$\hat{\beta}_F = \hat{\beta}_R = 0$$

(i. e., without taking sliding into account), then with $\hat{\beta}_F, \hat{\beta}_R$ given by the observer. In both cases, control gains equal to these used in simulation have been used, i. e., $(k_2, k_3) = (0.09, 0.6)$. The vehicle's longitudinal velocity during experiments was 3 m/s. The bottom part of Fig. 49.19 shows the tracking error along the path with the blue dashed line corresponding to using $(\beta_F, \beta_R) = (0, 0)$ in the control law, and the red dashed-dotted line corresponding to the use of sliding angles estimated values.

These experimental results confirm the simulation ones reported before. The improvement in tracking accuracy resulting from the use of estimated sliding angles in the control law is clearly illustrated: the lateral error remains small (smaller than 0.25 m) all the time and it becomes negligible as soon as the terrain's slope is alsmost constant (beyond the curvilinear abcissa 30 m). This is consistent with the observer's stability and convergence analysis when the sliding angles are assumed constant.

49.6 Complementary Issues and Bibliographical Guide

49.6.1 General Trailer Systems

Most of the control design approaches here presented and illustrated for unicycle-like and car-like vehicles can be extended to the case of trains of vehicles composed of trailers hitched to a leading vehicle. In particular, the methods of Sect. 49.4 which are specific to nonholonomic systems can be extended to this case, provided that the kinematic equations of motion of the system can be transformed (at least semiglobally) into a chained system [49.6]. This basically requires that the hitch point of each trailer is located on the rear-wheel axle of the preceding vehicle [49.22]. For instance, the transformation to the chained form is not possible when there are two (or more) successive trailers with off-axle hitch points [49.23]. So-called *general trailer systems* (with off-axle hitch points) raise difficult control design issues, and the literature devoted to them is sparse. For this reason, and also because these systems are not met in applications as frequently as simpler vehicles, control methods specifically developed for them are not reported here. Nonetheless, a few related references are given next. The path-following problem has been considered in, e.g., [49.24] for a system with two trailers and, more generally, in [49.25, Chap. 3] and [49.26] for an arbitrary number of trailers. To our knowledge, the problem of stabilizing nonstationary reference trajectories has not been addressed for these systems (except in the single trailer case for which the system can be transformed into the chained form [49.23, 27]). In fact, the explicit calculation of feasible trajectories joining a given configuration to another is already a very difficult problem, even in the absence of obstacles. As for the asymptotic stabilization of fixed configurations, the problem can (in theory) be solved by using existing general methods developed for the larger class of controllable driftless systems. However, the calculations associated with these methods quickly become intractable when the number of trailers increases. More specific and simpler ones have been proposed in [49.28], for an arbitrary number of trailers and the asymptotic stabilization of a reduced set of configurations, and in [49.29], in the case of two trailers and arbitrary fixed configurations.

Avidbots Ex1016
Page 1287 of 2154

49.6.2 Sensor-Based Motion Control

The control laws described in the present chapter, and their calculation, involve the online measurement, eventually complemented by the online estimation, of variables depending on the position of the vehicles in their environment. Measures can be acquired via the use of various sensors (odometry, GPS, proximetry, vision, etc.). Usually, various treatments are applied to raw sensory data prior to computing the control variables themselves. For instance, noise filtering and state estimation are such basic operations, well documented in the automatic-control literature. Among all sensors, vision sensors play a particularly important role in robotic applications, due to the richness and versatility of the information which they provide. The combination of visual data with feedback control is often referred to as *visual servoing*. In Chap. 34, a certain number of visual servoing tasks are addressed, mostly in the context of manipulation and/or under the assumption that realizing the robot task is equivalent to controlling the pose of a camera mounted on an omnidirectional manipulator. In a certain number of cases, the concepts and methods described in this chapter can be adapted, without much effort, to the context of mobile robots. These cases basically correspond to the control methods adapted from robotic manipulation which are described in Sect. 49.3 of the present chapter. For instance, automatic driving via the control of the visually estimated lateral distance between a robotic vehicle and the side of a road, or car-platooning by controlling the frontal and lateral distances to a leading vehicle, can be addressed with the control techniques reported in Chap. 34. The reason is that it is possible to simply recast these techniques in the form of the control laws proposed in Sect. 49.3. However, there are also vision-based applications for nonholonomic mobile robots which cannot be solved by applying classical visual-servoing techniques. This is the case, for instance, of the task objectives addressed in Sect. 49.4, an example of which is the stabilization of the complete posture (i. e., position and orientation) of a nonholonomic vehicle at a desired one. Vision-based control problems of this type have been addressed in [49.11, 30].

49.6.3 Sliding Effects and Other Dynamical Issues

Results of Sect. 49.5 can be viewed as a first step to addressing dynamical issues associated with sliding effects. The kinematic control models (49.53)–(49.54) can be extended to other wheeled vehicles ([49.31] for the mobile robot classification proposed by [49.32] and [49.33] for a trailer system). On-line estimation of sliding angles via an observer can also be achieved by exploiting the dynamics of these angles and making use of complementary measurements (from an IMU for example). A result in this direction is proposed in [49.21] where a model of sliding angle dynamics is used at the observer level, thus yielding a much more reactive estimation of these angles. Note that the kinematic control law of Sect. 49.5 may still be used in this case. When prior knowledge of the reference path is available, a predictive control strategy may also be adopted to reduce transient error effects associated with slow steering angle dynamics.

Various risks related to the vehicle's integrity when operating at high speed, like e.g., swing around or roll-over, were not addressed in this chapter. In [49.34] these issues are treated at the path planning level. Knowledge of the robot's dynamics can also be used to relate control inputs to a metrics measuring the importance of such risks in order to introduce constraints at the control level [49.35]. A multimodels approach (kinematic and dynamic) can be used to translate stability aspects into control constraints, for instance by defining a maximal admissible longitudinal velocity. A predictive control strategy may also help to limit the risk of roll-over, of steering saturation, or of swing around [49.36]. Finally, it is possible to take advantage of additional actuators, as in the case of four independent actuated wheels [49.37], or by using inclination actuators [49.38].

49.6.4 Bibliographical Guide

A few former surveys on the control of WMRs have been published. Let us mention [49.39–41], which contain chapters on the modeling and control issues. A detailed classification of kinematic and dynamic models for the different types of WMR structures, on which Chap. 24 is based, is provided in [49.32]. The use of the chain form to represent WMR equations has been proposed in [49.42], then generalized in [49.22].

Path following may have been the first mobile robot control problem addressed by researchers in robotics. Among the pioneering works, let us cite [49.43, 44]. Several results presented in the present chapter are based on [49.6, 45].

The problem of tracking admissible trajectories for unicycle-type and car-like vehicles is treated in the books [49.39–41], and also in numerous conference and journal papers. Several authors have addressed this problem by applying dynamic feedback linearization techniques. In this respect, one can consult [49.46–48], and [49.39, Chap. 8], for instance.

Numerous papers on the asymptotic stabilization of fixed configurations have been published. Among

them, [49.49] provides an early overview of feedback control techniques elaborated for this purpose, and also a list of references. The first result presenting a time-varying feedback solution to this problem, in the case of a unicycle-type vehicle, is in [49.4]. The conference paper [49.50] provides a survey on time-varying feedback stabilization, in the more general context of nonlinear control systems. More specific results, like Propositions 49.2 and 49.4, are given in [49.6, 7]. Other early results on the design of smooth time-varying feedbacks can be found in [49.51, 52], for example. Concerning continuous (but not Lipschitz-continuous) time-varying feedbacks yielding exponential convergence, one can consult [49.53]. Designs of hybrid discrete/continuous fixed-point stabilizers can be found in [49.54–57], for instance. Discontinuous control design techniques are not addressed in the present chapter, but the interested reader will find examples of such feedbacks in [49.58, 59].

To our knowledge, the control approach presented in Sect. 49.4.6, which is based on the concept of transverse functions [49.9, 10], is the first attempt to address the problem of tracking *arbitrary* trajectories (i. e., not necessarily feasible for the controlled robot). Implementation issues and experimental results for this approach can be found in [49.11, 12]. An overview of trajectory tracking problems for wheeled mobile robots, with a detailed case study of car-like systems, is presented in [49.13].

Video-References

VIDEO 181 Tracking of an admissible trajectory with a car-like vehicle
available from http://handbookofrobotics.org/view-chapter/49/videodetails/181

VIDEO 182 Tracking of arbitrary trajectories with a truck-like vehicle
available from http://handbookofrobotics.org/view-chapter/49/videodetails/182

VIDEO 243 Tracking of an omnidirectional frame with a unicycle-like robot
available from http://handbookofrobotics.org/view-chapter/49/videodetails/243

VIDEO 435 Mobile robot control in off-road condition and under high dynamics
available from http://handbookofrobotics.org/view-chapter/49/videodetails/435

References

49.1 M. Buehler, K. Iagnemma, S. Sanjiv (Eds.): *The 2005 DARPA Grand Challenge: The Great Robot Race*, Springer Tracts in Advanced Robotics, Vol. 36 (Springer, Berlin, Heidelberg 2007)

49.2 P. Morin, C. Samson: Motion control of wheeled mobile robots. In: *Springer Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008) pp. 799–826

49.3 R.W. Brockett: Asymptotic stability and feedback stabilization. In: *Differential Geometric Control Theory*, ed. by R.W. Brockett, R.S. Millman, H.J. Sussmann (Birkhäuser, Boston 1983)

49.4 C. Samson: Velocity and torque feedback control of a nonholonomic cart, Lect. Notes Control Inform. Sci. **162**, 125–151 (1991)

49.5 J.-M. Coron: Global asymptotic stabilization for controllable systems without drift, Math. Control Signals Syst. **5**, 295–312 (1992)

49.6 C. Samson: Control of chained systems. Application to path following and time-varying point-stabilization, IEEE Trans. Autom. Control **40**, 64–77 (1995)

49.7 P. Morin, C. Samson: Control of non-linear chained systems. From the Routh–Hurwitz stability criterion to time-varying exponential stabilizers, IEEE Trans. Autom. Control **45**, 141–146 (2000)

49.8 D.A. Lizárraga: Obstructions to the existence of universal stabilizers for smooth control systems, Math. Control Signals Syst. **16**, 255–277 (2004)

49.9 P. Morin, C. Samson: Practical stabilization of driftless systems on Lie groups: the transverse function approach, IEEE Trans. Autom. Control **48**, 1496–1508 (2003)

49.10 P. Morin, C. Samson: A characterization of the Lie algebra rank condition by transverse periodic functions, SIAM J. Control Optim. **40**(4), 1227–1249 (2001)

49.11 G. Artus, P. Morin, C. Samson: Control of a maneuvering mobile robot by transverse functions, Symp. Adv. Robot Kinemat. (ARK) (2004) pp. 459–468

49.12 G. Artus, P. Morin, C. Samson: Tracking of an omnidirectional target with a nonholonomic mobile robot, IEEE Conf. Adv. Robotics (ICAR) (2003) pp. 1468–1473

49.13 P. Morin, C. Samson: Trajectory tracking for nonholonomic vehicles: overview and case study, Proc. 4th Int. Workshop Robot Motion Control (RoMoCo), ed. by K. Kozlowski (2004) pp. 139–153

49.14 R. Lenain, B. Thuilot, C. Cariou, P. Martinet: High accuracy path tracking for vehicles in presence of sliding. application to farm vehicle automatic guidance for agricultural tasks, Auton. Robots **21**(1), 79–97 (2006)

Avidbots Ex1016
Page 1289 of 2154

49.15 T.D. Gillespie: *Fundamentals of Vehicle Dynamics* (SAE, Warrendale 1992)
49.16 P.R. Dahl: Solid friction damping of mechanical vibrations, AIAA J. **14**(12), 1675–1682 (1976)
49.17 C. Canudas de Wit, H. Olsson, K.J. Astrom, P. Lischinsky: A new model for control of systems with friction, IEEE Trans. Autom. Control **40**(3), 419–425 (1995)
49.18 C. Canudas de Wit, P. Tsiotras: Dynamic tire friction models for vehicle traction control, Proc. 38th IEEE Conf. Decis. Control, Vol. 4 (1999)
49.19 E. Bakker, L. Nyborg, H.B. Pacejka: Tyre modeling for use in vehicle dynamics studies, International Conference of the Society of Automotive Engineers (SAE) (1987) pp. 2190–2204
49.20 H.B. Pacejka: *Tyre and Vehicle Dynamics* (Butterworth-Heinemann, Oxford 2002)
49.21 R. Lenain, B. Thuilot, C. Cariou, P. Martinet: Mixed kinematic and dynamic sideslip angle observer for accurate control of fast off-road mobile robots, J. Field Robotics **27**(2), 181–196 (2010)
49.22 O.J. Sørdalen: Conversion of the kinematics of a car with *n* trailers into a chained form, IEEE Int. Conf. Robot. Autom. (ICRA) (1993) pp. 382–387
49.23 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness, motion planning and trailer systems, IEEE Int. Conf. Decis. Control (1993) pp. 2700–2705
49.24 P. Bolzern, R.M. DeSantis, A. Locatelli, D. Masciocchi: Path-tracking for articulated vehicles with off-axle hitching, IEEE Trans. Control Syst. Technol. **6**, 515–523 (1998)
49.25 D.A. Lizárraga: Contributions à la Stabilisation des Systèmes Non-Linéaires et à la Commande de Véhicules Sur Roues, Ph.D. Thesis (INRIA-INPG, University of Grenoble, Grenoble 2000)
49.26 C. Altafini: Path following with reduced off-tracking for multibody wheeled vehicles, IEEE Trans. Control Syst. Technol. **11**, 598–605 (2003)
49.27 F. Lamiraux, J.-P. Laumond: A practical approach to feedback control for a mobile robot with trailer, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 3291–3296
49.28 D.A. Lizárraga, P. Morin, C. Samson: Chained form approximation of a driftless system. Application to the exponential stabilization of the general N-trailer system, Int. J. Control **74**, 1612–1629 (2001)
49.29 M. Venditelli, G. Oriolo: Stabilization of the general two-trailer system, IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 1817–1823
49.30 M. Maya-Mendez, P. Morin, C. Samson: Control of a nonholonomic mobile robot via sensor-based target tracking and pose estimation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 5612–5618
49.31 D. Wang, C.B. Low: An analysis of wheeled mobile robots in the presence of skidding and slipping: Control design perspective, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2379–2384
49.32 G. Campion, G. Bastin, B. d'Andréa-Novel: Structural properties and classification of kynematic and dynamic models of wheeled mobile robots, IEEE Trans. Robotics Autom. **12**, 47–62 (1996)
49.33 C. Cariou, R. Lenain, M. Berducat, B. Thuilot: Autonomous maneuvers of a farm vehicle with a trailed implement in headland, Proc. 7th Int. Conf. Inform. Control Autom. Robotics (ICINCO), Vol. 2 (2010) pp. 109–114
49.34 K. Iagnemma, S. Shimoda, Z. Shiller: Near-optimal navigation of high speed mobile robots on uneven terrain, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 4098–4103
49.35 N. Bouton, R. Lenain, B. Thuilot, P. Martinet: A new device dedicated to autonomous mobile robot dynamic stability: application to an off-road mobile robot, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2010) pp. 3813–3818
49.36 O. Hach, R. Lenain, B. Thuilot, P. Martinet: Avoiding steering actuator saturation in off-road mobile robot path tracking via predictive velocity control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5523–5528
49.37 E. Lucet, C. Grand, D. Salle, P. Bidaud: Stabilization algorithm for a high speed car-like robot achieving steering maneuver, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 2540–2545
49.38 M. Krid, F. Ben-Amar: Design and control of an active anti-roll system for a fast rover, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 274–279
49.39 C. Canudas de Wit, B. Siciliano, G. Bastin (Eds.): *Theory of Robot Control* (Springer, Berlin, Heidelberg 1996)
49.40 J.-P. Laumond (Ed.): *Robot Motion Planning and Control*, Lecture Notes in Control and Information Sciences, Vol. 229 (Springer, Berlin, Heidelberg 1998)
49.41 Y.F. Zheng (Ed.): *Recent Trends in Mobile Robots*, World Scientific Series in Robotics and Automated Systems, Vol. 11 (World Scientific, Singapore 1993)
49.42 R.M. Murray, S.S. Sastry: Steering nonholonomic systems in chained form, IEEE Int. Conf. Decis. Control (1991) pp. 1121–1126
49.43 E.D. Dickmanns, A. Zapp: Autonomous high speed road vehicle guidance by computer vision, Proc. IFAC 10th World Congr. Autom. Control. (1987)
49.44 W.L. Nelson, I.J. Cox: Local path control for an autonomous vehicle, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1998) pp. 1504–1510
49.45 C. Samson: Path following and time-varying feedback stabilization of a wheeled mobile robot, Proc. Int. Conf. Autom. Robotics Comput. Vis. (1992)
49.46 B. d'Andréa-Novel, G. Campion, G. Bastin: Control of nonholonomic wheeled mobile robots by state feedback linearization, Int. J. Robotics Res. **14**, 543–559 (1995)
49.47 A. De Luca, M.D. Di Benedetto: Control of nonholonomic systems via dynamic compensation, Kybernetica **29**, 593–608 (1993)
49.48 M. Fliess, J. Lévine, P. Martin, P. Rouchon: Flatness and defect of non-linear systems: Introductory

Avidbots Ex1016
Page 1290 of 2154

theory and examples, Int. J. Control **61**, 1327–1361 (1995)
49.49 I. Kolmanovsky, N.H. McClamroch: Developments in nonholonomic control problems, IEEE Control Syst. **15**, 20–36 (1995)
49.50 P. Morin, J.-B. Pomet, C. Samson: Developments in time-varying feedback stabilization of nonlinear systems, IFAC Nonlinear Control Syst. Design Symp. (1998) pp. 587–594
49.51 J.-B. Pomet: Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift, Syst. Control Lett. **18**, 467–473 (1992)
49.52 A.R. Teel, R.M. Murray, G. Walsh: Nonholonomic control systems: from steering to stabilization with sinusoids, Int. J. Control **62**, 849–870 (1995)
49.53 R.T. M'Closkey, R.M. Murray: Exponential stabilization of driftless nonlinear control systems using homogeneous feedback, IEEE Trans. Autom. Control **42**, 614–6128 (1997)
49.54 M.K. Bennani, P. Rouchon: Robust stabilization of flat and chained systems, Eur. Control Conf. (1995) pp. 2642–2646
49.55 P. Lucibello, G. Oriolo: Stabilization via iterative state feedback with application to chained-form systems, IEEE Conf. Decis. Control (1996) pp. 2614–2619
49.56 O.J. Sørdalen, O. Egeland: Exponential stabilization of nonholonomic chained systems, IEEE Trans. Autom. Control **40**, 35–49 (1995)
49.57 P. Morin, C. Samson: Exponential stabilization of nonlinear driftless systems with robustness to unmodeled dynamics, ESAIM Control Optim. Calc. Var. **4**, 1–36 (1999)
49.58 A. Astolfi: Discontinuous control of nonholonomic systems, Syst. Control Lett. **27**, 37–45 (1996)
49.59 C. Canudas de Wit, O.J. Sørdalen: Exponential stabilization of mobile robots with nonholonomic constraints, IEEE Trans. Autom. Control **37**(11), 1791–1797 (1992)

Avidbots Ex1016
Page 1291 of 2154

Multimedia Contents

50. Modeling and Control of Robots on Rough Terrain

Keiji Nagatani, Genya Ishigami, Yoshito Okada

In this chapter, we introduce modeling and control for wheeled mobile robots and tracked vehicles. The target environment is rough terrains, which includes both deformable soil and heaps of rubble. Therefore, the topics are roughly divided into two categories, wheeled robots on deformable soil and tracked vehicles on heaps of rubble.

After providing an overview of this area in Sect. 50.1, a modeling method of wheeled robots on a deformable terrain is introduced in Sect. 50.2. It is based on terramechanics, which is the study focusing on the mechanical properties of natural rough terrain and its response to off-road vehicle, specifically the interaction between wheel/track and soil. In Sect. 50.3, the control of wheeled robots is introduced. A wheeled robot often experiences wheel slippage as well as its sideslip while traversing rough terrain. Therefore, the basic approach in this section is to compensate the slip via steering and driving maneuvers. In the case of navigation on heaps of rubble, tracked vehicles have much advantage. To improve traversability in such challenging environments, some tracked vehicles are equipped with subtracks, and one kinematical modeling method of tracked vehicle on rough terrain is introduced in Sect. 50.4. In addition, stability analysis of such vehicles is introduced in Sect. 50.5. Based on such kinematical model and stability analysis, a sensor-based control of tracked vehicle on rough terrain is introduced in Sect. 50.6. Sect. 50.7 summarizes this chapter.

50.1 **Overview** 1268
- 50.1.1 Modeling Deformable Terrain for Robots 1268
- 50.1.2 Controlling Robots in Deformable Terrain 1268
- 50.1.3 Modeling and Controlling Robots in Heaps of Rubble 1269

50.2 **Modeling of Wheeled Robot in Rough Terrain** 1270
- 50.2.1 Dynamics of Mobile Robots 1270
- 50.2.2 Wheel–Terrain Interaction Mechanics 1271
- 50.2.3 Wheel Traction Performance 1273
- 50.2.4 Model Uncertainty and Soil Parameter Identification 1273

50.3 **Control of Wheeled Robot in Rough Terrain** 1274
- 50.3.1 Slip-Compensated Path Follower 1274
- 50.3.2 Steering and Driving Maneuvers 1275

50.4 **Modeling of Tracked Vehicle on Rough Terrain** 1276
- 50.4.1 Parameterization of a Track 1277
- 50.4.2 Contact with a Single Point in the Terrain Surface 1277
- 50.4.3 Contact with the Terrain Surface Represented by a Point Cloud 1277

50.5 **Stability Analysis of Tracked Vehicles** 1278

50.6 **Control of Tracked Vehicle on Rough Terrain** 1279
- 50.6.1 Ground Detection and Trimming of the Scanned Data 1280
- 50.6.2 Determination of Desired Posture 1280
- 50.6.3 Determination of Desired Subtrack Positions 1280
- 50.6.4 Stability Evaluation of Desired Pose 1281
- 50.6.5 Stabilization of Desired Pose 1281
- 50.6.6 Position Control of Subtracks 1281

50.7 **Summary** 1281

Video-References 1281

References 1282

Avidbots Ex1016
Page 1292 of 2154

50.1 Overview

50.1.1 Modeling Deformable Terrain for Robots

The category of rough terrain includes a variety of ground conditions. In the case of weak and fine gravel, the motion of a wheeled mobile robot becomes relatively complicated because the interaction mechanics of the wheel in a rough terrain differs from that of an indoor robot on a flat surface; a wheel on gravel has multiple point contact, and a wheel on loose soil has a surface contact that deforms the soil around the wheel.

There has been a large amount of research on robot mobility analysis in the military community [50.1, 2]. These studies have primarily focused on empirical analyses of large vehicles (i. e., gross vehicle weights of several tons). The mobility analyses of small mobile robots that consider the interaction mechanics of a slipping wheel on rough terrain have also been investigated. Additionally, a multibody system simulation for the longitudinal slip of tires with respect to the tire–soil interaction has been demonstrated [50.3]. A dynamic interaction model, denoted as soil contact model (SCM), has been developed to provide the dynamics of a plastically deformable surface using an elevation grid, as well as to consider the multipass effect of wheels, shown in Fig. 50.1 [50.4]. Recently, multibody dynamics toolkit, such as Vortex (VIDEO 184), can be used to simulate terrain reaction forces for mobility analyses of mobile robots [50.5].

Recently, terramechanics [50.6] has been widely used in applications such as off-road UGVs (unmanned ground vehicles) [50.7, 8] and planetary rovers on Mars. A fundamental requirement for such off-road mobile robots is to maintain a traversing capability in rough terrain composed of loose sand, mud, pebbles, etc. One typical issue in such an environment is wheel slippage, which is generated because of the soil deformation at the wheel–terrain contact patch. Wheel slippage often degrades the mobility performance of the robot because some amount of tractive power from a driving actuator is consumed by the soil deformation. Furthermore, the wheel slippage induces vehicle sideslip, which is an additional disturbance for the motion control of a wheeled robot. Figure 50.2 shows one typical example of lateral- and side-slip situation of a wheeled mobile robot in rough terrain. One typical example of terramechanics-based modeling of rough terrain is described in Sect. 50.2. Recently, a study of terradynamics of legged locomotion was proposed for traversal in granular media in [50.9] and is shown in VIDEO 186.

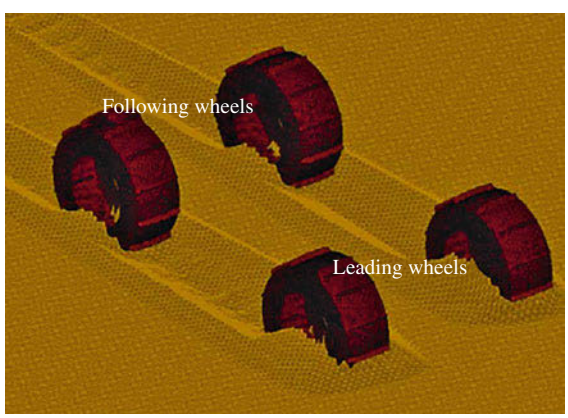


Fig. 50.1 Dynamics of a plastically deformable surface in consideration with the multipass effect of wheels (after [50.4])

50.1.2 Controlling Robots in Deformable Terrain

A wheeled robot often experiences wheel slippage as well as vehicle sideslip while traversing rough terrain. This degrades the motion control performance of the robot, i. e., a robot could deviate from the path to be followed or a wheel may get trapped in loose soil. Therefore, a control scheme that can derive the appropriate steering/driving maneuvers is necessary for a wheeled robot to achieve traversability on a rough terrain and to manage any slippage problems that may arise.

A large number of studies related to the path-following control of mobile robots have been published. General information on path-following issues can be obtained from [50.10–12]. *Rezaei* et al. investigated an online path-following strategy combined with a simultaneous localization and mapping (SLAM) algorithm for a car-like robot in outdoor environments [50.13].



Fig. 50.2 Lateral- and side-slip situation of wheeled mobile robot

Avidbots Ex1016
Page 1293 of 2154

Coelho and *Nunes* et al. proposed a Kalman-based active observer controller for the path-following of wheeled mobile robots [50.14]. *Helmick* et al. developed a path-following algorithm that included slip compensation by using visual odometry and a Kalman filter [50.15, 16]. The method was applied to the rocker-bogie configuration with six steerable wheels robot, shown in Fig. 50.3. In Sect. 50.3, a path-following control scheme for a four-wheeled mobile robot is introduced. The experimental validation of the control described in the section was reported in [50.17].

50.1.3 Modeling and Controlling Robots in Heaps of Rubble

In environments that involve heaps of rubble, such as an urban search and rescue situation, the approach for modeling and controlling robots is different from the afore mentioned methodology. To tackle such challenging environments, tracks are used to enhance the mobility of a robot in such environments because they have more contact with the terrain than wheels. Some tracked vehicles are equipped with active subtracks, and multitracks configuration enables the vehicle to get over relatively large steps. A typical geometrical model of tracked vehicles with subtracks is introduced in Sect. 50.4.

In case that the target rubble is stable, kinematic approaches are useful for navigation and control of the robot. In this case, stability analysis is very important. In classical stability analysis, the energy stability margin (ESM) proposed by *Messuri* and *Klein* was defined in terms of the potential energy of the robot [50.18]. Hirose assumed that increasing the robot's weight did not always contribute to its stability because it also increased the dynamic disturbance around its center of gravity. Therefore, he proposed the normalized ESM (NESM), in which the robot's weight was normalized [50.19]. For this criterion, the NESM is used to evaluate the stability of a robot on the basis of the vertical distance between the initial and largest heights of the center of gravity during a rollover. Recently, *Tubouchi*'s group proposed a method to control tracked vehicles exposed to random steps [50.20] (Fig. 50.4). An artificial uneven environment is random steps, arranged with different lengths of squared timbers. It also appears in VIDEO 189. This environment is used to evaluate the performance for urban search and rescue robots [50.21]. The basic objective is to maintain the maximal stability of the robot at every step of its path in the random step field. The earlier-given topic relating to stability analysis is introduced in Sect. 50.5. Another approach for stability analysis of tracked vehicles-based NESM was reported in [50.22].

To evaluate the stability of vehicles, terrain detection is an important technology. Therefore, many robotics researchers have focused on the terrain mapping, particularly based on the SLAM, described in Chap. 46, and machine learning. LIDAR (light detection and ranging) and stereo cameras are widely used for the terrain mapping (Sect. 22.3). *Vandapel* et al. proposed a classification method of three-dimensional (3-D) point cloud data on the basis of its geometrical features [50.23], and *Lacaze* et al. demonstrated a method of finding a path in heavy grasses [50.25]. To improve the LIDAR-based mapping, *Ohno* et al. proposed a method to obtain thin objects such as nets, poles, and wires [50.26]. Figure 50.5 shows a mapping result at an outdoor rough terrain.

If the rubble is unstable, modeling the machine-rubble interaction is very complicated. One challenging



Fig. 50.3 Rocky 8: rocker-bogie configuration with six steerable wheels robot

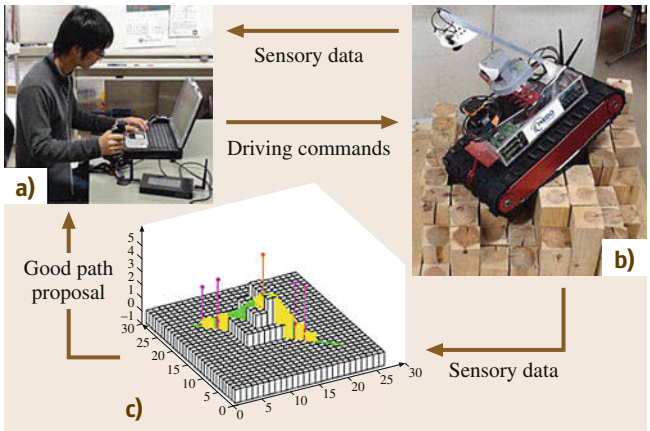


Fig.50.4a–c Teleoperated system of tracked vehicle. Itconsists of **(a)** a remote operation station, **(b)** a mobile robot platform, and **(c)** path planner that maintain the maximal stability of the robot (after [50.20])

Avidbots Ex1016
Page 1294 of 2154

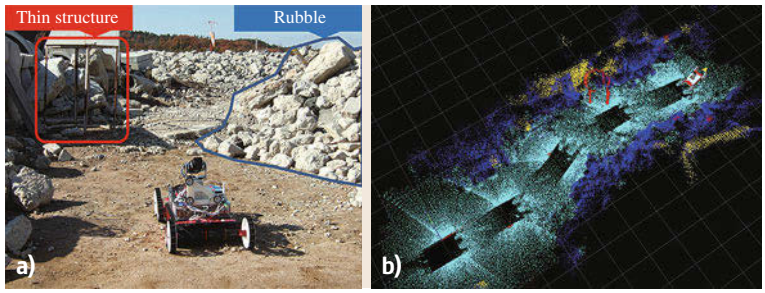


Fig.50.5a,b Classification of outdoor rubble data. **(a)** Target field: Hyogo Prefectural Emergency Management and Training Center, and **(b)** classified result for traversal (after [50.23])

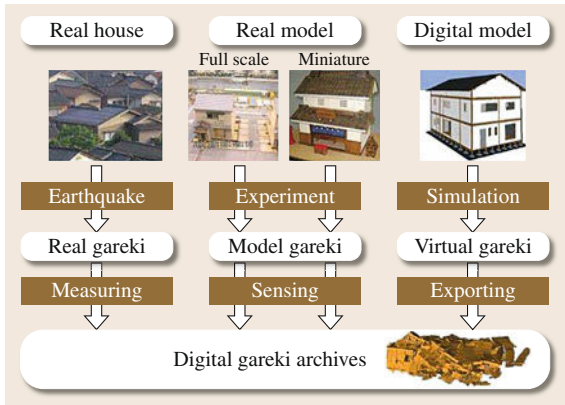


Fig. 50.6 Approaches for gareki data acquisition (after [50.24])

approach to modeling rubble is called gareki (rubble in Japanese) engineering [50.27, 28]. The model includes not only the structure data, but also the internal force of each gareki element. To obtain the data, they proposed to use information about the following, as shown in Fig. 50.6: (1) a real-collapsed building, (2) artificial full- and miniature-scale collapsed buildings, and (3) a simulated virtual digital building. Their research aims to eventually apply the gareki model for the control of mobile robots and rescue missions.

The sensor-based method is another approach for controlling robots in unstable rubbles. Such robots often have multiple tracks connected by joints to maneuver over natural/artificial steps, bumps, and stairs. Some are equipped with passive joints [50.29], but many are equipped with active joints [50.30]. Multitrack configurations enable the vehicle to maneuver over a step that is higher than the radius of the tracks. To enable such motion, a sensor-based method is proposed to control the multitracks. The robot moves and detects the surface shape of the ground continuously, and it changes the configuration of the subtracks based on the surface information. Once the surface of the ground is deformed by the traversal of the robot, the subtracks adapt quickly. A representative example for the control of robots in unstable rubble is shown in Sect. 50.6. Another sensor-based approach is to use force-feedback from contact points. *Inoue* et al., embedded force sensors in their subtracks to obtain the contact force, and changed the configuration of the subtracks [50.31].

50.2 Modeling of Wheeled Robot in Rough Terrain

This section introduces a mechanics of wheeled mobile robot in rough terrain, particularly focusing on a modeling of wheel–terrain interaction.

50.2.1 Dynamics of Mobile Robots

Figure 50.7 depicts a schematic illustration of a dynamic model for a wheeled mobile robot. The robot is modeled as an articulated multibody system. The equation of motion for the robot is numerically given by

$$\mathbf{H}\begin{pmatrix}\dot{\mathcal{V}}_\mathrm{b}\\ \ddot{\boldsymbol{q}}\end{pmatrix}+\boldsymbol{C}+\boldsymbol{G}=\begin{pmatrix}\mathcal{F}_\mathrm{b}\\ \boldsymbol{\tau}\end{pmatrix}+\mathbf{J}^\mathrm{T}\mathcal{F}_\mathrm{e}\,, \tag{50.1}$$

where the symbols have the following meanings: $\mathbf{H}$: inertia matrices composed of multibody of the robot, $\boldsymbol{C}$: velocity depending term, $\boldsymbol{G}$: gravity term, $\mathcal{V}_\mathrm{b}$: translational and angular velocities of the vehicle, $\boldsymbol{q}$: joint angles (such as wheel rotation and steering angles), $\mathcal{F}_\mathrm{b}$: forces and moments at the centroid of the vehicle body, $\boldsymbol{\tau}$: torques acting at each joint (driving/steering torques), $\mathbf{J}$: Jacobian matrix, $\mathcal{F}_\mathrm{e}$: external forces and moments acting at the centroid of each wheel, namely

$$f_{ij}(i=\{r,l\},j=\{r,m,f\})\,.$$

The dynamic motion of the robot with given traveling and steering maneuvers are numerically obtained

by successively solving (50.1). Here, a key approach for the dynamic model is to incorporate a well-defined contact model into the equation of motion in order to calculate the external forces and torques at the contact patch between the wheel and terrain. Some recent works have reported dynamic simulation tools combined with the terramechanics wheel model: for example, [50.32, 33] for the NASA's rovers, and [50.4, 34] for the ExoMars.

50.2.2 Wheel–Terrain Interaction Mechanics

Terramechanics is a study focusing on the mechanical properties of natural rough terrain and its response to off-road vehicle, specifically the interaction between wheel/track and soil. In 1960s, *Bekker* developed a classical terramechanics with many of the key concepts related to vehicle–terrain response, such as a well-known pressure-sinkage equation and shear stress model [50.35, 36]. Wong developed a comprehensive procedure for predicting the performance of both driven and towed wheels [50.37–39]. The procedure calculates the wheel mechanics with applying the stress distribution model beneath the wheel.

Terramechanics-based approach for vehicle–terrain mechanics includes mainly three methodologies [50.40, 41]: (1) Analytical approach, (2) empirical approach, and (3) numerical approach.

The analytical approach is based on a theoretical model for vehicle–terrain interactions along with experimental model validation. The empirical method is to utilize a practical measurement of soil strength

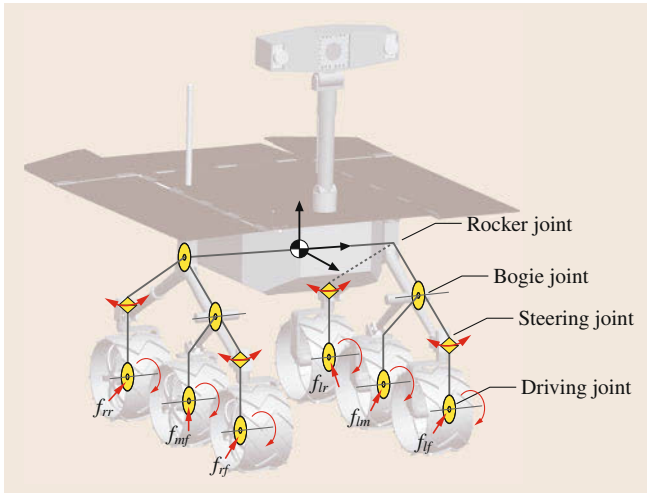


Fig. 50.7 Dynamic model of wheeled mobile robot

by a specialized apparatus in order to estimate vehicle traversability on weak/rough terrain. The numerical method includes the finite element method (FEM) and discrete element method (DEM) that models soils as vast numbers of multiple particles and simulates each particle's behavior while accounting vehicle–terrain interaction [50.42–44].

This subsection focuses on the analytical method and introduces a typical interaction model of rigid wheel on deformable terrain (Fig. 50.8). It should be noted that Chap. 55.3.13 also addresses the wheel traction mechanics in detail, and therefore, the following subsection recalls some key equations in order to dis-

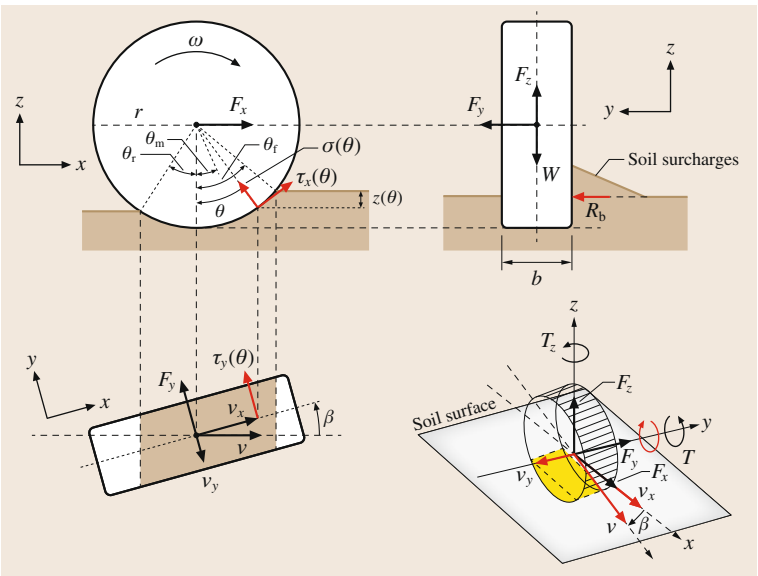


Fig. 50.8 Wheel–terrain contact model

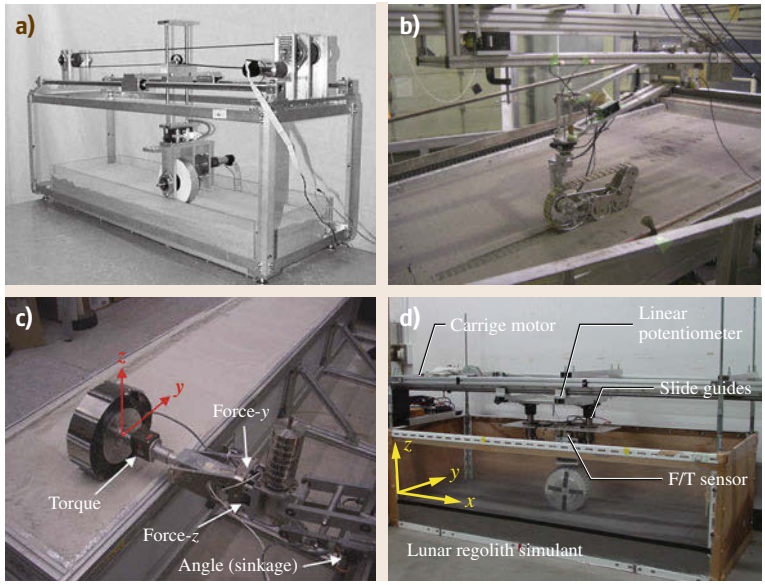


Fig.50.9a–d Single wheel test beds for experimental validation of terramechanics models. **(a)** Single wheel test bed at MIT (after [50.45]), **(b)** Single track test bed at JAXA (after [50.46]), **(c)** Single wheel test bed at DLR (after [50.47]), **(d)** Single wheel test bed at Tohoku University. (after [50.48])

cuss a wheel traction characteristics on rough terrain, soil parameter identification, and model uncertainty.

Wheel Slippage

Wheel slippage is generally observed when a robot travels on loose soil. Also, the wheel will slip in the lateral direction while the robot steering or traversing on sloped terrain. The slip ratio (i. e., slip in the longitudinal direction of wheel travel) is defined as a function of the longitudinal traveling velocity of the wheel v_x and the circumferential velocity of the wheel $r\omega$, where r is the wheel radius and ω represents the angular velocity of the wheel [50.37]

$$s = \begin{cases} (r\omega - v_x)/r\omega & (|r\omega| \geq |v_x| : \text{driving}) , \\ (r\omega - v_x)/v_x & (|r\omega| < |v_x| : \text{braking}) . \end{cases} \tag{50.2}$$

The slip ratio assumes a value in the range from -1 to 1.

The slip in the lateral direction of the wheel is expressed by the slip angle β, which is defined as a function of v_x and the lateral traveling velocity v_y, as

$$\beta = \begin{cases} \tan^{-1}(v_y/v_x) & (v_x \neq 0) , \\ \text{sgn}(v_y) \cdot \pi/2 & (v_x = 0) . \end{cases} \tag{50.3}$$

Wheel Traction Forces

The wheel–terrain contact forces, including the drawbar pull F_x, side force F_y, vertical force F_z, and resistance torque T, are calculated as the integral of the normal and shear stresses from the entry angle θ_f and the exit angle θ_r [50.38, 48]

$$F_x = rb \int_{\theta_r}^{\theta_f} \{\tau_x(\theta) \cos\theta - \sigma(\theta) \sin\theta\} \, d\theta , \tag{50.4}$$

$$F_y = \int_{\theta_r}^{\theta_f} \left[rb\tau_y(\theta) + R_b \{r - z(\theta) \cos\theta\}\right] d\theta , \tag{50.5}$$

$$F_z = rb \int_{\theta_r}^{\theta_f} \{\tau_x(\theta) \sin\theta + \sigma(\theta) \cos\theta\} \, d\theta , \tag{50.6}$$

$$T_x = r^2 b \int_{\theta_r}^{\theta_f} \tau_x(\theta) \, d\theta , \tag{50.7}$$

where $\sigma(\theta)$ is the normal stress, $\tau_x(\theta)$ and $\tau_y(\theta)$ are the shear stresses in the longitudinal or lateral direction. $z(\theta)$ is the wheel sinkage at an angle of θ, b is the wheel width, and R_b is the bulldozing resistance generated on a side wall of the wheel.

Experimental Validation

An experimental validation of the afore mentioned wheel–terrain interaction model is performed by the use of a single wheel test bed (Fig. 50.9), with varied state parameters such as soil or wheel slip conditions. A carriage velocity controlled relative to a wheel velocity realizes a wheel slip (or traction load) while measuring wheel traction forces, wheel

sinkage, and others. Experimental data obtained from the test bed are then compared with the values obtained from numerical simulation of the wheel traction model.

50.2.3 Wheel Traction Performance

The wheel–terrain model can be exploited for an evaluation of the mobility performance of a wheeled robot (i. e., traversability on sloped or deformable terrain [50.49, 50]). Such mobility evaluation technique is further valuable for the mobility system design in order to find a feasible wheel/track design that maximizes the traction performance for off-road locomotion under some specific constraints [50.46, 47, 51].

Ishigami et al. proposed a wheel traction diagram called a thrust–cornering characteristic diagram that determines the slope traversal criteria with respect to wheel slip conditions [50.17]). Figure 50.10 illustrates the relationship between the thrust and cornering forces of wheel on rough terrain, with varied slip ratios/slip angles. The thrust and cornering forces F_c and F_T are obtained by calculating the drawbar pull and side force

$$F_c = F_x \sin\beta + F_y \cos\beta \,, \tag{50.8}$$

$$F_T = F_x \cos\beta - F_y \sin\beta \,. \tag{50.9}$$

Considering the slope traversal scenario as shown in Fig. 50.11, the thrust–cornering characteristic diagram can draw the criteria with a minimum cornering force and thrust force required for a slope traversal: Case 1 as a traversal and uphill traveling, Case 2 as a straight-line traversal, and Case 3 as a traversal and downhill traveling. In addition, a robot decelerates or cannot move forward due to its negative thrust force. This diagram, for example, indicates that the slip ratio must exceed 0.6 in order to generate positive thrust force, and also, the cornering force varies primarily with the slip angle. Therefore, the diagram can determine the slope traversability on the basis of the corresponding slippage, indicated by several characteristic curves on the diagram.

50.2.4 Model Uncertainty and Soil Parameter Identification

It should be noted that the classical terramechanics model has primarily developed for an application for large-heavy vehicles (hundreds/thousands kg weight). Therefore, several assumptions that may cause an inaccurate calculation of wheel traction performance of the classical model would be omitted while exploiting the classical model to an analysis of small lightweight robot, resulting in an inaccurate calculation of wheel traction performance. The error due to the inadequate assumptions for the small-lightweight robot can be addressed by the following approaches: model improvement and uncertainty analysis of model parameters. It should be noted that the Bekker's pressure-sinkage model is assumed that the contact patch of wheel on deformable soil (circumferential section) is discretized

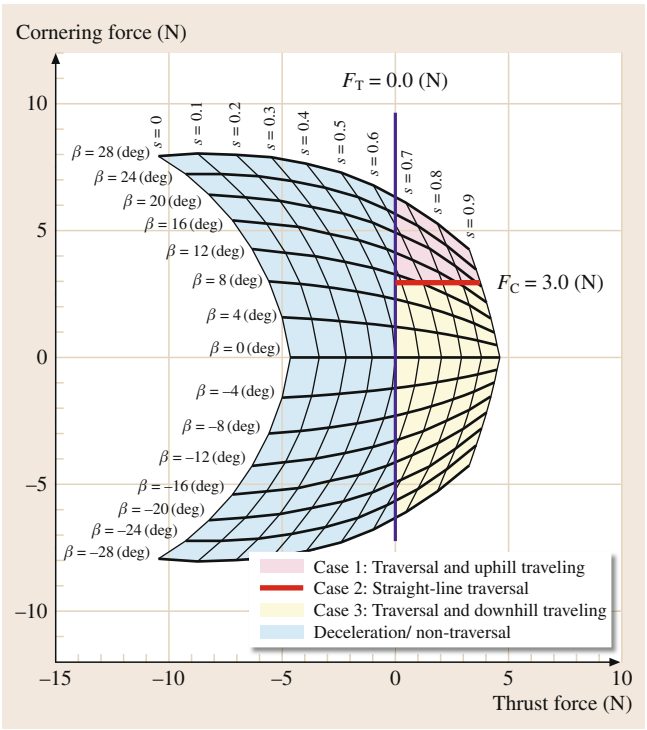


Fig. 50.10 Thrust–cornering characteristic diagram

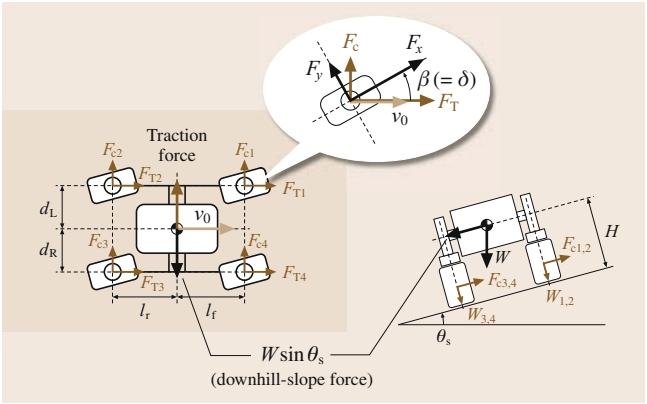


Fig. 50.11 Force balance on slope traversal

Avidbots Ex1016
Page 1298 of 2154

as a series of consecutive flat plate; however, *Bekker* noted [50.36]:

> *Predictions for wheels smaller than 20 inches in diameter become less accurate as wheel diameter decreases, because the sharp curvature of the loading area was neither considered in its entirety nor is it reflected in bevameter tests.*

Also another research reported that the assumptions provide inaccurate prediction for vehicles with wheels diameter less than about 50 cm (approx.) and normal loading less than 45 N (approx.) [50.36].

Some researchers have tried to update/improve the classical terramechanics model that can be successfully applied to relatively light weight vehicles: For example, *Nagatani* et al. developed a direct measurement device for the normal stress distribution [50.52]. A wheel diameter dependent pressure-sinkage model has been proposed by [50.53]. *Senatore* and *Iagnemma* proposed an improved approach for the calculation of shear deformation modulus [50.54].

Model parameters in the wheel–terrain interaction related to the soil properties are subject to uncertainties since their values would stochastically vary with location. Several researches have addressed the soil parameter identification: an on-line terrain parameter estimator with simplifying classical terramechanics equations was proposed by [50.55]; a modified nonlinear wheel–terrain interaction model for an identification of pressure-sinkage coefficient, internal friction angle, and shear deformation modulus [50.56]. Some recent works have also attempted to predict rover mobility even in uncertain conditions: a learning-based approach for slip-prediction and traversability analysis of a wheeled robot [50.57]; and a statistical method for mobility prediction of a robot with taking into account of terrain uncertainty [50.58].

50.3 Control of Wheeled Robot in Rough Terrain

This section introduces a control scheme for the wheeled robot that compensates wheel-and-vehicle slippages while following a given path is described.

A wheeled robot often experiences wheel slippages as well as vehicle sideslip while traversing in rough terrain, which degrades a performance of a motion control of the robot: i. e., a robot will deviate from a path to be followed, or a wheel might get stuck into loose soil. Therefore, a control scheme that can derive appropriate steering/driving maneuvers is necessary for a wheeled robot to achieve a rough terrain traverse and to manage any slippage problems.

A significant number of studies related to the path following control of mobile robot have been published. A general note for path following issues can be seen in [50.10–12]. *Rezaei* et al. investigated an online path following strategy combined with a simultaneous localization and mapping (SLAM) algorithm for a car-like robot in outdoor environments [50.13]. *Coelho* and *Nunes* et al. proposed a Kalman-based active observer controller for the path following of wheeled mobile robots [50.14].

As noted before, a key issue of the path following on rough terrain is to compensate wheel-and-vehicle slippage. This subsection briefly introduces a slip-compensated path follower along with a derivation for steering and driving maneuvers for path following [50.15–17] which is shown in VIDEO 188.

50.3.1 Slip-Compensated Path Follower

A two-dimensional path-following problem of a mobile robot is shown in Fig. 50.12. Here, a path following control generally performs to let the vehicle velocity v_0 coincide with the tangent of a given path such that the distance and orientation errors decrease to zero. However, once the vehicle experiences a sideslip with a certain amount of slip angle β_0, the vehicle has an additional orientation error of β_0 even

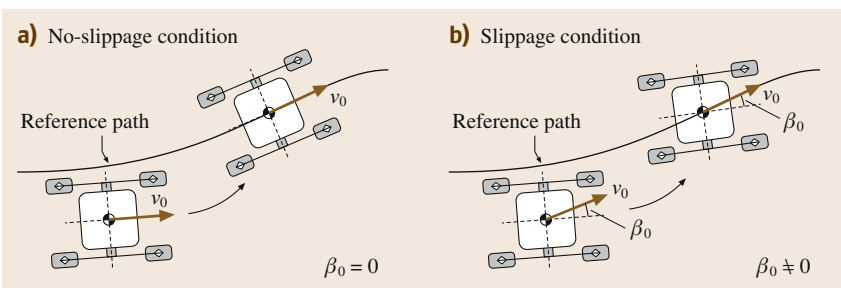


Fig.50.12a,b Path following problem without **(a)** and with **(b)** slip angle of vehicle

Avidbots Ex1016
Page 1299 of 2154

when the vector of vehicle velocity completely follows the path (Fig. 50.12b).

The path follower basically derives a commanded robot velocity vector $(\dot{x}_{\mathrm{cmd}}, \dot{y}_{\mathrm{cmd}}, \dot{\theta}_{\mathrm{cmd}})$ while the robot position (x_0, y_0, θ_0) and a reference path vector $(x_{\mathrm{p}}, y_{\mathrm{p}})$ are given. The slip-compensated path follower described here consists of two algorithms to achieve a path following task through high-slip environments.

First, the carrot heading algorithm determines the heading error of the robot θ_{e}, which is an angle calculated from the intersection of a circle centered on the robot coordinate with the desired path (Fig. 50.13):

$$\theta_{\mathrm{e}} = \theta_{\mathrm{d}} - \theta_0 \tag{50.10}$$

Here, θ_{d} is the desired heading angle which is a tangent angle on the intersection point and θ_0 is the robot heading angle. A large radius of the circle will generate a smooth motion of the robot while neglect a small feature of the path. A small radius will reduce the total path following error, but requires frequent heading changes for small path errors.

Once the heading error is determined, the slip-compensated path following algorithm executes the heading controller that calculates the commanded yaw rate of the robot $\dot{\theta}_{\mathrm{cmd}}$ from the heading error θ_{e} and the yaw slip rate $\dot{\beta}_0$

$$\dot{\theta}_{\mathrm{cmd}} = \frac{\left(K_1 \cdot \theta_{\mathrm{e}} + K_2 \cdot \dot{\beta}_0\right)}{T_{\mathrm{s}}}\,, \tag{50.11}$$

where K_1 and K_2 are the tuned controller gains, and T_{s} is the sampling period.

Note that the command for the linear velocity $\dot{x}_{\mathrm{cmd}}$ and $\dot{y}_{\mathrm{cmd}}$ are closely related to the maximum speed of the driving motors and the desired heading angle $\dot{\theta}_{\mathrm{cmd}}$. Practically, these commands are accomplished by actuators of the robotic mobility system, namely steering and driving mechanisms. The following subsection describes the derivation of those steering and driving maneuvers that satisfies the heading controller command $\dot{\theta}_{\mathrm{cmd}}$ as defined in (50.11).

50.3.2 Steering and Driving Maneuvers

Here, a nonholonomic kinematic model of a four-wheeled mobile robot with each wheel being independently steerable is considered. Figure 50.14 illustrates a two-dimensional kinematic model of the mobile robot having the slip angle of the vehicle β_0 and lateral wheel slippage β_i (the subscript i denotes the wheel identification number, $i = 1, \ldots, 4$ in this case). The dimension of the rover is defined by l_{f}, l_{r}, d_{R}, and d_{L}. The position and orientation of the vehicle are defined as $(x_0,\ y_0,\ \theta_0)$, and the position of each wheel is defined as $(x_i,\ y_i)$. In this model, the following assumptions are considered: the distance between the wheels is constant, and the steering axle of each wheel is perpendicular to the terrain surface.

The nonholonomic constraints of the mobile robot are defined by the following equations, with taking the lateral slips of wheels and that of vehicle into account

$$\begin{aligned} \dot{x}_0 \sin\phi_0 - \dot{y}_0 \cos\phi_0 &= 0\,, \\ \dot{x}_i \sin\phi_i - \dot{y}_i \cos\phi_i &= 0\,, \end{aligned} \tag{50.12}$$

where $\phi_0 = \theta_0 + \beta_0$ and $\phi_i = \theta_0 + \delta_i + \beta_i$. The geometric constraints between each wheel and the centroid of

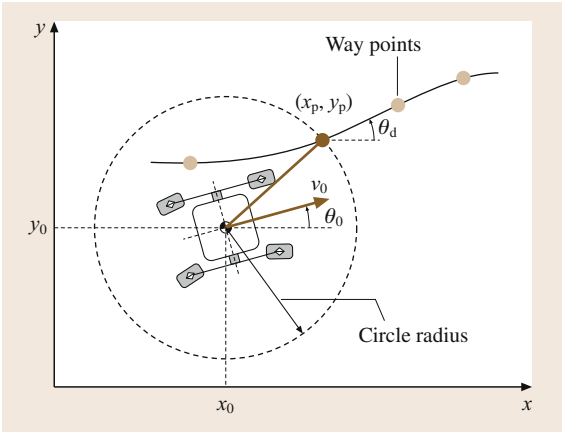


Fig. 50.13 Path following with carrot heading calculation

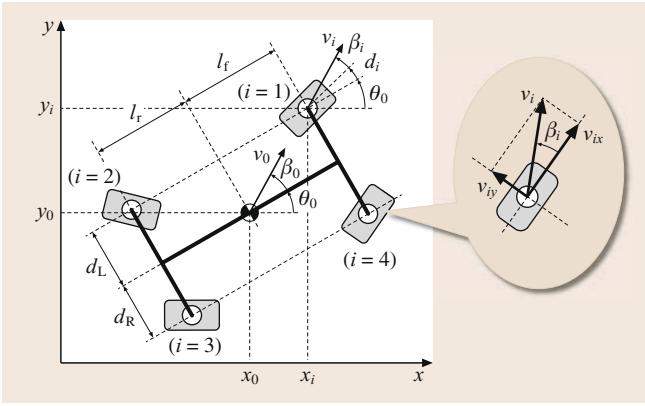


Fig. 50.14 Kinematic model of four-wheeled rover with wheel/vehicle slips

the vehicle are written as

$$\left.\begin{aligned} x_1 &= x_0 + l_\mathrm{f}\cos\theta_0 - d_\mathrm{L}\sin\theta_0 \\ x_2 &= x_0 - l_\mathrm{r}\cos\theta_0 - d_\mathrm{L}\sin\theta_0 \\ x_3 &= x_0 - l_\mathrm{r}\cos\theta_0 + d_\mathrm{R}\sin\theta_0 \\ x_4 &= x_0 + l_\mathrm{f}\cos\theta_0 + d_\mathrm{R}\sin\theta_0 \end{aligned}\right\} \rightarrow x_i = x_0 + X_i \,, \tag{50.13}$$

$$\left.\begin{aligned} y_1 &= y_0 + l_\mathrm{f}\sin\theta_0 + d_\mathrm{L}\cos\theta_0 \\ y_2 &= y_0 - l_\mathrm{r}\sin\theta_0 + d_\mathrm{L}\cos\theta_0 \\ y_3 &= y_0 - l_\mathrm{r}\sin\theta_0 - d_\mathrm{R}\cos\theta_0 \\ y_4 &= y_0 + l_\mathrm{f}\sin\theta_0 - d_\mathrm{R}\cos\theta_0 \end{aligned}\right\} \rightarrow y_i = y_0 + Y_i \,. \tag{50.14}$$

Given the desired heading angle $\theta_0 = \theta_\mathrm{d}$ and desired linear velocity v_d, the desired steering angle $\delta_{\mathrm{d}i}$ that satisfies the path following control as well as the slip compensation is obtained as follows: first, transform (50.12)

$$\delta_{\mathrm{d}i} = \tan^{-1}(\dot{y}_i/\dot{x}_i) - \theta_\mathrm{d} - \beta_i \,. \tag{50.15}$$

Subsequently, substituting (50.13) and (50.14) into (50.15), the desired steering angle is determined as follows

$$\delta_{\mathrm{d}i} = \tan^{-1}\left(\frac{v_\mathrm{d}\sin\theta_\mathrm{d} - \dot{Y}_i(\dot{\theta}_\mathrm{d})}{v_\mathrm{d}\cos\theta_\mathrm{d} - \dot{X}_i(\dot{\theta}_\mathrm{cmd})}\right) - \theta_\mathrm{d} - \beta_i \,. \tag{50.16}$$

The desired yaw rate $\dot{\theta}_\mathrm{cmd}$ is given from (50.11).

The driving maneuver is practically executed by the wheel angular velocity of the robot. From Fig. 50.14, the wheel linear velocity v_i is given as

$$v_i = r\omega_i/\cos\beta_i \,, \tag{50.17}$$

v_i is also expressed by $\dot{x}_i$ or $\dot{y}_i$

$$v_i = \dot{x}_i/\cos\phi_i = \dot{y}_i/\sin\phi_i \,. \tag{50.18}$$

Substituting (50.13) or (50.14) into the above equations, the desired wheel angular velocity $\omega_{\mathrm{d}i}$ can be obtained

$$\omega_{\mathrm{d}i} = \begin{cases} \left(v_\mathrm{d}\cos\theta_\mathrm{d} + \dot{X}_i(\dot{\theta}_\mathrm{cmd})\right)\cos\beta_i/r\cos\phi_i \\ \qquad (\theta_\mathrm{d} \le \pi/4) \,, \\ \left(v_\mathrm{d}\sin\theta_\mathrm{d} + \dot{Y}_i(\dot{\theta}_\mathrm{cmd})\right)\cos\beta_i/r\sin\phi_i \\ \qquad (\theta_\mathrm{d} \ge \pi/4) \,. \end{cases} \tag{50.19}$$

Position and orientation errors relative to a desired path should be accurately determined for deriving the control input as well as calculating the corresponding steering and driving maneuvers. A visual odometry method is of promising technique for such accurate measurement of position, orientation, velocity, and slippage, and also it is sufficiently robust for loose sandy terrain.

The visual odometry estimates the traveling velocity of the vehicle based on the optical flow vectors obtained from the time-consecutive images taken by an onboard camera(s). Integrating the velocity estimates with inertial measurement unit (IMU) readouts or stereo images for pose estimation provides an accurate estimation of the six degrees of freedom of the rover motion.

50.4 Modeling of Tracked Vehicle on Rough Terrain

Tracks are one of typical mechanism that can enhance the mobility of a robot on rough terrain because they can have contacts between the terrain more widely than wheels. Thus, many types of robots which are required to maneuver rough terrain such as military and search-and-rescue robots have been equipped with tracks.

Some tracked vehicles often have multiple tracks connected by joints to maneuver natural/artificial steps, bumps and stairs. Some are equipped with passive joints, but many are equipped with active joints. Multitracks configuration enables the vehicle to get over a step which is higher than the radius of the tracks.

It should be noted that many tracked vehicles are intended to maneuver the unknown rough terrain consisted of rubbles, rocks, concrete, and woods. Because such types of terrain are nonuniform unlike fine soil discussed in the Sect. 50.2, to dynamicaly model them is obviously difficult. In other words, we can easily sense the shape of the terrain, but dynamic parameters as well as center of gravity (COG), weight or stiffness of terrain cluster.

Thus, to geometrically consider the interaction between tracks and terrain is a reasonable approach. In this section, a geometrical model of a contact between a track and the terrain surface is introduced. The analysis includes (1) a parameterization of a generic shaped track, (2) contacts between the track and a single point in the terrain surface, and (3) contacts between the track and the terrain surface represented by a point cloud.

Avidbots Ex1016
Page 1301 of 2154

50.4.1 Parameterization of a Track

The geometry of a generic-shaped track is the key issue that should be discussed for modeling a tracked vehicle on rough terrain. In this section, we introduce a parameterization of the shape of a generic track.

As shown in Fig. 50.15, a generic track can be parameterized by the radius of the first and the second pulleys, R and r respectively, and the distance between the centers of the pulleys L.

In addition, let us embed the coordinate frame; its origin is at the center of the adjacent track and the x axis is along the center line of the adjacent track. The relative pose of the track respect to the adjacent track can be parameterized by the distance between the origin and the center of the first pulley, and the angle of the joint that connects the tracks θ.

50.4.2 Contact with a Single Point in the Terrain Surface

To discuss contacts between a track and a single point in the terrain surface is a good entrance to understand it between a track and the whole terrain surface. In the following sections, we geometrically describe the former contacts with the parameters of the track defined in the previous section.

As shown in Fig. 50.16, a track generally has four sections that can contact with the terrain surface; since it is tracked, it comprises a round and straight sections. Moreover, it can be in two states, namely, folded and spread states. When the track is in the spaded state, side-S supports the robot; on the other hand, when it is in the folded state, side-F is in contact with the terrain. We call the case where side-S is in contact with the ground as the *spreading mode* and the case where side-F is in contact with the ground as the *folding mode*.

In addition, we have assumed the pose of the track to be 0[°] when its straight section on side-S is in contact with the flat surface; we have also assumed the direction in which the track lifts its second pulley from 0[°] to be positive.

The geometrical relationship between the contact point and pose of the track is summarized in Table 50.1. For example, in the case of the spreading mode and the contact on the straight section (Fig. 50.15), we can derive the following equation according to the geometry of the track contacting with the surface

$$\begin{aligned}\theta_{\text{contact}} &= \theta_1 + \theta_2 \\ &= \tan^{-1}\frac{z}{x - x_{\text{support}}} \\ &\quad + \sin^{-1}\frac{r}{\sqrt{(x - x_{\text{support}})^2 + z^2}} \,. \end{aligned} \tag{50.20}$$

50.4.3 Contact with the Terrain Surface Represented by a Point Cloud

Now we are well prepared to geometrically understand contacts between the track and the terrain surface. In this section, we describe those contacts by iterating a point cloud that represents the terrain surface.

Let the shape of the terrain surface be represented by a set of points $\{u_1, u_2, \ldots, u_n\}$. Then, the contact angle of the track with the surface is determined using the following equation

$$\theta_{ref} = \begin{cases} \min(\theta_{\text{contact},1}, \ldots, \theta_{\text{contact},n}): \\ \quad \text{in spreading mode} \,, \\ \max(\theta_{\text{contact},1}, \ldots, \theta_{\text{contact},n}): \\ \quad \text{in folding mode} \,. \end{cases} \tag{50.21}$$

To calculate a contact angle of the track with a single point in the terrain surface, we need to determine which section and which side of the track makes the contact.

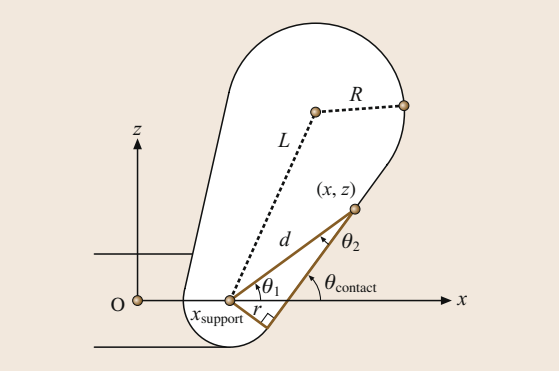


Fig. 50.15 Example of geometrical analysis contact between track and single point on the terrain surface

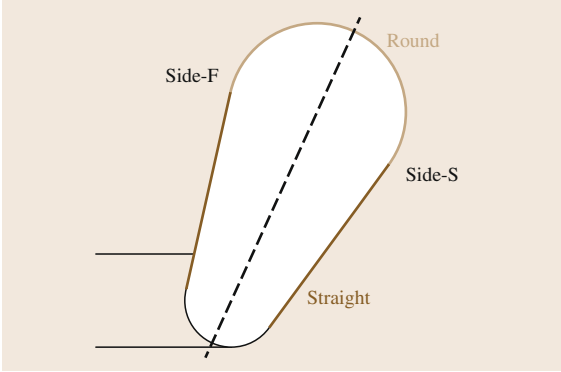


Fig. 50.16 Generic-shaped track consisted of staraight and round sections

Part E | 50.4

Table 50.1 Geometric-based contact angle of subtrack

Section		θ_{contact}
Spreading	Straight	$\tan^{-1}\dfrac{z}{x-x_{\text{support}}}+\sin^{-1}\dfrac{r}{\sqrt{(x-x_{\text{support}})^2+z^2}}$
	Round	$\cos^{-1}\dfrac{d^2+L^2-R^2}{2Ld}+\tan^{-1}\dfrac{z}{x-x_{\text{support}}}-\sin^{-1}\dfrac{R-r}{L}$
Folding	Straight	$\tan^{-1}\dfrac{z}{x-x_{\text{support}}}-\sin^{-1}\dfrac{r}{\sqrt{(x-x_{\text{support}})^2+z^2}}-2\sin^{-1}\dfrac{R-r}{L}$
	Round	$\tan^{-1}\dfrac{z}{x-x_{\text{support}}}-\sin^{-1}\dfrac{R-r}{L}-\cos^{-1}\dfrac{d^2+L^2-R^2}{2Ld}$

The type of the section can be determined by checking the distance between the center of the first pulley and the point; if $d < \sqrt{L^2+R^2}$ the straight section makes the contact, else the round section does. On the other hand, the side will be determined by the choice of the application.

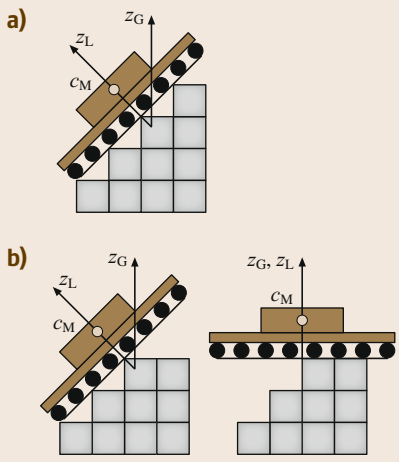
Now we understand a geometric model which describes contacts between the tracked vehicle consisted of multiple tracks and the free-formed terrain surface whose shape is represented as a point cloud. In the following subsection, we introduce an application example utilizing the contact model.

50.5 Stability Analysis of Tracked Vehicles

As shown in the previous sections, it is hardly possible to build an indention model of a rough terrain consisted of rubbles. Thus, in this research domain, stability criteria based only on geometrical parameters are often used to evaluate pose of the tracked vehicle on the rough terrain.

This section introduces two typical stability criteria which have been adopted for tracked vehicles; (1) support polygon and (2) normalized energy stability margin.

The support polygon is a simple and reasonable criterion to determine whether an object on a surface is statically stable. The support polygon for the tracked vehicle is equal to the horizontal projection of the convex hull of all the contact points between the tracks and the terrain. With this support polygon, we can consider



Label	Balance quality	Technical details
Red	Turn upside down or get stuck	Pitch > π/4 or roll > π/6 or at least one of conditions A is not satisfied
Orange	Lose balance on purpose	Two optional postures exist
Magenta	Climb up or slide down	Oscillations in posture estimation algorithm
Cyan	Jump down	Jump of CM between two stable postures < 50 mm
Yellow	Fair	NESM parameter < 1
Green	Good	NESM parameter ≥ 1

Fig.50.17a,b Example of stability analysis of tracked vehicle on a rough terrain based on support polygon and NESM (after [50.20]); **(a)** Stable posture corresponding to green label in table. **(b)** Unstable posture corresponding to orange label in table

Avidbots Ex1016
Page 1303 of 2154

that the vehicle is statically stable when it contains the projection of the vehicle's center of gravity.

Another stability criterion is the normalized energy stability margin, or NESM, proposed by Hirose. The NESM is a criterion that is used to evaluate the stability of a robot on the basis of the vertical distance between the initial position of its center of gravity and its highest position during tumbling. Although it is mainly used for walking robots, its evaluation only requires the positions of contact points with the ground and the center of gravity of the robot. In other words, there is no conceptual difference if this criterion is applied to the stability of tracked vehicles.

One typical application of stability criteria for the tracked vehicle is to rank possible path in a navigation. For example, Magid proposed a rating method for possible paths based on stability analysis utilizing the support polygon and NESM. This method evaluates the stability of each adjacent grid of the current position of the vehicle; paths to grids on which the support polygon condition is violated are denied. Then survived paths are ranked utilizing the NESM (Fig. 50.17).

50.6 Control of Tracked Vehicle on Rough Terrain

In this section, an approach for the autonomous control of active joints in a tracked vehicle with multiple tracks is introduced. As shown in the previous subsections, the geometric model of the tracked vehicle gives the pose of each track, which makes contact with the rough terrain on the basis of the shape of the terrain surface near the vehicle. The autonomous control utilizes the geometric model to determine the desired angles of the active joints between the tracks to perform a smooth maneuver on an unknown rough terrain.

Tracked vehicles with multiple tracks and joints typically have better locomotion than single-tracked vehicles. This is because one can change the pose of a track by actuating an adequate joint to negotiate bumps or steps in a rough terrain. Some multitracked vehicles a are equipped with a single-degree-of-freedom (1-DOF) tracked arms, often called subtracks, and the others are snake shaped.

However, multitracked vehicles are difficult to control manually because of their multiple degrees of freedom. For military and search-and-rescue robots in particular, which are likely to be teleoperated on an unknown rough terrain with limited camera views, it is quite difficult to control all of their DOFs manually.

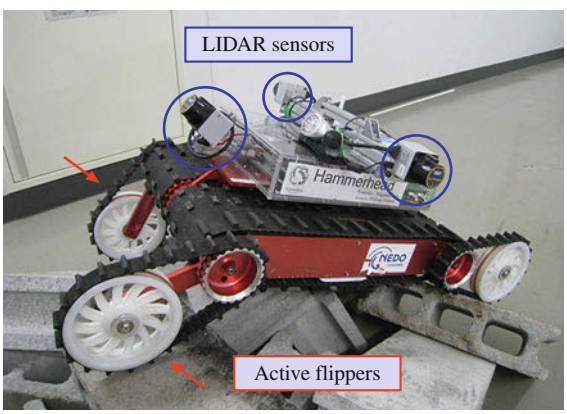


Fig. 50.18 Tracked vehicle Kenaf equipped with four subtracks enhansing mobility and three LIDAR sensors sensing shape of terrain

In the rest of this section, an approach for the autonomous control of active subtracks on the basis of the geometric model of a multitracked vehicle is introduced. In the control algorithm, the geometric model automatically determines the desired poses of active joints, which support subtracks on the basis of the terrain shape obtained in real time.

The multitracked vehicle Kenaf, which has autonomous subtrack control, is shown in Fig. 50.18. Kenaf has two main tracks covering its main body and four subtracks actuated by 1-DOF joints on each corner of the main body. It has 6-DOFs in total: 4-DOFs corresponding to joints supporting the subtracks, 1-DOF corresponding to the one main track and two subtrack belts on the right, and 1-DOF corresponding to the one main track and two subtrack belts on the left.

In addition, Kenaf is equipped with three LIDAR sensors, one on the front, one on the right side, and one on the left side of the main body, to obtain data on the shape of the terrain online. Scans from LIDARs are integrated to estimate the three-dimensional shape of the terrain near the vehicle. The aforementioned approach for autonomous control of active subtracks are described in [50.59].

Figure 50.19 shows the flowchart of the algorithm for the autonomous subtrack control, and its actual motions can be seen in VIDEO 190 and VIDEO 191. The control algorithm is divided into six steps and summarized as follows:

1. Slices of the shape of the terrain around the robot are first obtained from the LIDAR sensors attached

to the robot body and the three-dimensional terrain shape near the robot is estimated.
2. The desired posture (roll and pitch angles) of the body is then calculated on the basis of the estimated terrain shape.
3. The desired positions of the subtracks that realize the desired posture of the robot body are also determined.
4. Next, the stability of the desired posture and subtrack positions is evaluated.
5. If the desired pose (posture of the robot body and subtrack positions) are unstable, then the desired pose is redefined and steps (3)–(5) are repeated.
6. When the desired subtrack positions that realize a stable posture are generated, position control of the subtracks is finally performed.

In the following sections, we explain each step of the algorithm.

50.6.1 Ground Detection and Trimming of the Scanned Data

In this step, we obtain the scanned points U_{target} near the robot using the LIDAR sensors attached to the robot; ground detection is performed according to the approach described in the paper [50.59]. We first obtain two-dimensional terrain slices from three LIDAR sensors and integrate them according to each tagged and estimated position of the robot to generate the three-dimensional information of the terrain shape. At the end of this step, we filter distant points from the integrated terrain shape.

$$U_{\text{target}} = \{u_1, u_2, \ldots, u_m\} \tag{50.22}$$

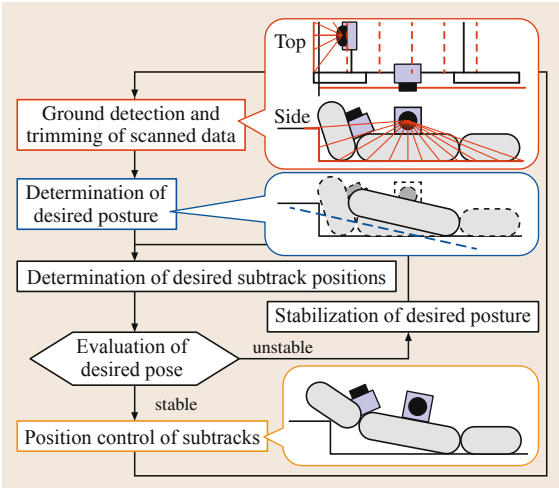


Fig. 50.19 Algorithm for autonomous control of subtracks

50.6.2 Determination of Desired Posture

We then determine the desired posture on the basis of the least-squares plane for the target terrain U_{target}. This is intended to enable smooth locomotion on rough terrain. Parameters a, b, and c of the least-squares plane $z = ax + by + c$ are determined using the following equations (an over-bar indicates the average of m scanned points in U_{target})

$$a = \frac{\alpha_{z,x}\alpha_{y,y} - \alpha_{x,y}\alpha_{y,z}}{\alpha_{x,x}\alpha_{y,y} - \alpha_{x,y}\alpha_{x,y}} , \tag{50.23}$$

$$b = \frac{\alpha_{y,z}\alpha_{x,x} - \alpha_{x,y}\alpha_{z,x}}{\alpha_{x,x}\alpha_{y,y} - \alpha_{x,y}\alpha_{x,y}} , \tag{50.24}$$

$$c = \overline{z_v} - \overline{x_v}a - \overline{y_v}b , \tag{50.25}$$

$$\alpha_{x,y} = \overline{x_u \cdot y_u} - \overline{x_u} \cdot \overline{y_u} , \tag{50.26}$$

$$\alpha_{y,z} = \overline{y_u \cdot z_u} - \overline{y_u} \cdot \overline{z_u} , \tag{50.27}$$

$$\alpha_{z,x} = \overline{z_u \cdot x_u} - \overline{z_u} \cdot \overline{x_u} , \tag{50.28}$$

$$\alpha_{x,x} = \overline{x_u \cdot x_u} - \overline{x_u} \cdot \overline{x_u} , \tag{50.29}$$

$$\alpha_{y,y} = \overline{y_u \cdot y_u} - \overline{y_u} \cdot \overline{y_u} . \tag{50.30}$$

Then, a translation to the coordinate system, if the robot body is parallel to the least-squares plane of the ground surface and makes contact with the ground, is described by the following equation in quaternion algebra:

$$\begin{pmatrix} 0 \\ x_{u_i'} \\ y_{u_i'} \\ z_{u_i'} \end{pmatrix} = q' \times \begin{pmatrix} 0 \\ x_{u_i} \\ y_{u_i} \\ z_{u_i} \end{pmatrix} \times q'^{-1} - \begin{pmatrix} 0 \\ 0 \\ 0 \\ \max(z_{u'}) \end{pmatrix} , \tag{50.31}$$

$$q' = \begin{pmatrix} \cos\dfrac{\theta_{\text{rot}}}{2} \\ b\sin\dfrac{\theta_{\text{rot}}}{2} \\ -a\sin\dfrac{\theta_{\text{rot}}}{2} \\ 0 \end{pmatrix} , \tag{50.32}$$

$$\theta_{\text{rot}} = \cos^{-1}\frac{1}{\sqrt{a^2 + b^2 + 1}} . \tag{50.33}$$

50.6.3 Determination of Desired Subtrack Positions

In this step, using the desired body posture and the integrated terrain slices, we determine the desired subtrack positions on the basis of the geometric model of contacts between the vehicle and the terrain surface.

Avidbots Ex1016
Page 1305 of 2154

Our application allows the operator to manually switch between the spreading and the folding control modes. Each control mode employs a corresponding geometric calculation method described in Sect. 50.4 In other words, the autonomous controller uses the corresponding calculation to determine the desired subtrack positions according to the control mode specified by the operator.

In both modes, we determine the desired subtrack positions to make contact with the ground through the desired robot posture. In particular, we calculate the angular position of each subtrack that makes contact with each point scanned by the left and the right LIDAR sensors. The desired position of each subtrack is determined for the maximum or minimum angular position of the subtrack in the spreading or folding mode, respectively.

50.6.4 Stability Evaluation of Desired Pose

In the proposed controller, we have adopted the normalized energy stability margin (NESM) [50.19] as the stability criterion for the desired pose. The stability of a desired pose is evaluated by NESM, and the pose is redefined if the stability is insufficient.

In the case of a tracked vehicle with four subtracks, four contact points (front-right, front-left, rear-right, and rear-left) can be determined by the step described in Sect. 50.6.3. In addition, the four axes of tumbling that pass through the front, rear, right, and left contact points can be assumed. Hence, the stability of a tracked vehicle with four subtracks can be determined from the minimum value of the NESM about these four axes.

50.6.5 Stabilization of Desired Pose

When the NESM of the robot is less than a predetermined threshold, we repeat the following steps until a desired, stable pose is realized. This step is intended to realize the strategy 4:

1a. If the NESM about the front or rear is adopted, reduce the pitch angle of the desired posture to zero.
1b. When the NESM about the right or left is adopted, reduce the roll angle of the desired posture to zero.
2. Redefine the desired subtrack positions by recalculating them to realize the redefined desired posture.
3. Evaluate the NESM about the redefined posture and positions of subtracks.

50.6.6 Position Control of Subtracks

Finally, we perform position control of the subtracks to realize the desired subtrack position produced through the above-mentioned steps on the basis of the strategy described in the previous subsection.

All subtracks of Kenaf and Quince are controlled by microprocessors present on the built-in motor drivers. Each desired subtrack position is sent to the corresponding microprocessor as a reference for position control using the conventional proportional–integral–derivative (PID) controller.

50.7 Summary

In this chapter, an overview of the modeling and control of robots on the rough terrain motion has been provided. The target environment was rough terrains, which includes a variety of ground conditions. In the case of weak and fine gravel, the motion of a wheeled mobile robot becomes relatively complicated because the interaction mechanics of the wheel in rough terrains differs greatly from that of an indoor robot on a flat surface. In this chapter, we introduced a terramechanics-based model for the interactions between wheels and deformable terrains. Furthermore, we explained the steering control of wheeled mobile robots to enable path-following in such environments. In the case of an environment with heaps of rubble, the approach for controlling robots is different from the terramechanics approach. Therefore, we introduced an initial attempt at a sensor-based approach that involved measuring the terrain shape.

Video-References

VIDEO 184 Mobility prediction of rovers on soft terrain
available from http://handbookofrobotics.org/view-chapter/50/videodetails/184

VIDEO 185 Experiments of wheeled rovers in a sandbox covered with loose soil
available from http://handbookofrobotics.org/view-chapter/50/videodetails/185

Avidbots Ex1016
Page 1306 of 2154

VIDEO 186 Terradynamics of legged locomotion for traversal in granular media
available from http://handbookofrobotics.org/view-chapter/50/videodetails/186
VIDEO 187 Interaction human-robot supervision, long range science rover for Mars exploration
available from http://handbookofrobotics.org/view-chapter/50/videodetails/187
VIDEO 188 A path-following control scheme for a four-wheeled mobile robot
available from http://handbookofrobotics.org/view-chapter/50/videodetails/188
VIDEO 189 Evaluation test of tracked vehicles on random step fields in the Disaster City
available from http://handbookofrobotics.org/view-chapter/50/videodetails/189
VIDEO 190 Autonomous sub-tracks control
available from http://handbookofrobotics.org/view-chapter/50/videodetails/190
VIDEO 191 Autonomous sub-tracks control
available from http://handbookofrobotics.org/view-chapter/50/videodetails/191

References

50.1 M. Jurkat, C. Nuttall, P. Haley: *The AMC' 74 Mobility Model, Tech. Rep. 11921* (US Army Tank Automotive Command, Warren, 1975)
50.2 R.B. Ahlvin, P.W. Haley: *NATO Reference Mobility Model Edition II, NRMM User's Guide, Tech. Rep. GL-92-19* (US Army WES, Vicksburg, 1992)
50.3 A. Gibbesch, B. Schäfer: Multibody system modelling and simulation of planetary rover mobility on soft terrain, 8th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS), Munich (2005)
50.4 R. Krenn, A. Gibbesch, G. Hirzinger: Contact dynamics simulation of rover locomotion, Proc. 9th Int. Symp. on Artif. Intell., Robotics Autom. Space, Los Angeles (2007)
50.5 D. Holz, A. Azimi, M. Teichmann, J. Kövecses: Mobility prediction of rovers on soft terrain: Effects of wheel- and tool-induced terrain deformations, Proc. 15th Int. Conf. Climbing Walk. Robots Support Technol. Mob. Mach. (CLAWAR) (2012)
50.6 J.Y. Wong: *Theory of Ground Vehicles* (Wiley, New York 1978)
50.7 M. Buehler, K. Iagnemma, S. Singh (Eds.): *The 2005 DARPA Grand Challenge: The Great Robot Race Springer Tracts Adv. Robotics Ser*, Vol. 36 (Springer, Berlin, Heidelberg 2005)
50.8 M. Buehler, K. Iagnemma, S. Singh (Eds.): *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, Springer Tracts Adv. Robotics, Vol. 56 (Springer, Berlin, Heidelberg 2009)
50.9 C. Li, T. Zhang, D.I. Goldman: A terradynamics of legged locomotion on granular media, Science **339**, 1408–1412 (2013)
50.10 C. de Wit, H. Khennouf, C. Samson, O. Sordalen: Nonlinear control design for mobile robots. In: *Recent Trends in Mobile Robots*, World Scientific Series in Robotics and Automated System, Vol. 11, ed. by Y. Zheng (World Scientific, Singapore 1993)
50.11 A. Luca, G. Oriolo, C. Samson: Feedback control of nonholonomic car-like robots. In: *Robot Motion Planning and Control*, ed. by J. Laumond (Springer, Berlin, Heidelberg 1998) pp. 171–254
50.12 F. Rio, G. Jimenez, J. Sevillano, S. Vicente, A. Balcells: A generalization of path following for mobile robots, Proc. 1999 IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 7–12
50.13 S. Rezaei, J. Guivant, E. Nebot: Car-like robot path following in large unstructured environments, Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 2468–2473
50.14 P. Coelho, U. Nunes: Path-following control of mobile robots in presence of uncertainties, IEEE Trans. Robotics **21**(2), 252–261 (2005)
50.15 D. Helmick, Y. Cheng, D. Clouse, L. Matthies, S. Roumeliotis: Path following using visual odometry for a Mars rover in high-slip environments, Proc. 2004 IEEE Aerosp. Conf., Big Sky (2004) pp. 772–789
50.16 D. Helmick, S. Roumeliotis, Y. Cheng, D. Clouse, M. Bajracharya, L. Matthies: Slip-compensated path following for planetary exploration rovers, Adv. Robotics **20**(11), 1257–1280 (2006)
50.17 G. Ishigami, K. Nagatani, K. Yoshida: Slope traversal controls for planetary exploration rover on sandy terrain, J. Field Robotics **26**(3), 264–286 (2009)
50.18 D.A. Messuri, C.A. Klein: Automatic body regulation for maintaining stability of a legged vehicle during rough-terrain locomotion, IEEE J. Robotics Autom. **1**(3), 132–141 (1985)
50.19 S. Hirose, H. Tsukagoshi, K. Yoneda: Normalized energy stability margin and its contour of walking vehicles on rough terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 181–186
50.20 E. Magid, T. Tsubouchi, E. Koyanagi, T. Yoshida, S. Tadokoro: Controlled balance losing in random step environment for path planning of a teleoperated crawler-type vehicle, J. Field Robotics **28**(6), 932–949 (2011)
50.21 A. Jacoff, E. Messina, B.A. Weiss, S. Tadokoro, Y. Nakagawa: Test arenas and performance metrics for urban search and rescue robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas (2003) pp. 3396–3403
50.22 K. Ohno, V. Chun, T. Yuzawa, E. Takeuchi, S. Tadokoro, T. Yoshida, E. Koyanagi: Rollover avoidance using a stability margin for a tracked vehicle with sub-tracks, IEEE Int. Workshop Saf. Sec. Rescue Robotics (2009)
50.23 N. Vandapel, D. Huber, A. Kapuria, M. Hebert: Natural terrain classification using 3-D ladar data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 5 (2004) pp. 5117–5122

Avidbots Ex1016
Page 1307 of 2154

50.24 M. Onosato, S. Yamamoto, M. Kawajiri, F. Tanaka: Digital gareki archives: An approach to know more about collapsed houses for supporting search and rescue activities, IEEE Int. Symp. Saf. Secur. Rescue Robotics (SSRR) (2012) pp. 1–6

50.25 A. Lacaze, K. Murphy, M. Del Giorno: Autonomous mobility for the demo III experimental unmanned vehicles, AUVS Int. Conf. Unnanned Veh. (2002)

50.26 K. Ohno, T. Suzuki, K. Higashi, M. Tsubota, E. Takeuchi, S. Tadokoro: Classification of 3-D point cloud data that includes line and frame objects on the basis of geometrical features and the pass rate of laser rays, Proc. 8th Int. Conf. Field Serv. Robotics (2012)

50.27 M. Onosato, T. Watasue: Two attempts at linking robots with disaster information: InfoBalloon and gareki engineering, Adv. Robotics **16**(6), 545–548 (2002)

50.28 M. Onosato: Digital GAREKI modeling for exploring knowledge of disaster-collapsed houses, IEEE Int. Workshop Saf. Secur. Rescue Robotics (SSRR) (2006)

50.29 L. Woosub, K. Sungchul, K. Munsang, P. Mignon: ROBHAZ-DT3: Teleoperated mobile platform with passively adaptive double-track for hazardous environment applications, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 33–38

50.30 B. Yamauchi: Packbot: A versatile platform for military robotics, Proc. SPIE **5422**, 228–237 (2004)

50.31 D. Inoue, K. Ohno, S. Nakamura, S. Tadokoro, E. Koyanagi: Whole-body touch sensors for tracked mobile robots using force-sensitive chain guides, IEEE Int. Workshop Saf. Secur. Rescue Robotics (SSRR) (2008) pp. 71–76

50.32 A. Jain, J. Balaram, J. Cameron, J. Guineau, C. Lim, M. Pornerantz, G. Sohl: Recent developments in the ROAMS planetary rover simulation environment, Proc. 2004 IEEE Aerosp. Conf., Big Sky (2004) pp. 861–876

50.33 K. Iagnemma, C. Senatore, B. Trease, R. Arvidson, A. Shaw, F. Zhou, L. Van Dyke, R. Lindemann: Terramechanics modeling of mars surface exploration rovers for simulation and parameter estimation, ASME Int. Des. Eng. Tech. Conf. (2011)

50.34 R. Bauer, W. Leung, T. Barfoot: Development of a dynamic simulation tool for the exomars rover, Proc. 8th Int. Symp. Artif. Intell., Robotics Autom. Space, Munich (2005)

50.35 M.G. Bekker: *Theory of Land Locomotion* (Univ. Michigan Press, Ann Arbor 1956)

50.36 M.G. Bekker: *Introduction to Terrain-Vehicle Systems* (Univ. Michigan Press, Ann Arbor 1969)

50.37 J.Y. Wong: *Theory of Ground Vehicles*, 4th edn. (Wiley, Hoboken 2008)

50.38 J.Y. Wong, A.R. Reece: Prediction of rigid wheel performance based on the analysis of soil-wheel stresses – Part I: Performance of driven rigid wheels, J. Terramechanics **4**(1), 81–98 (1967)

50.39 J.Y. Wong, A.R. Reece: Prediction of rigid wheel performance based on the analysis of soil-wheel stresses – Part II: Performance of towed rigid wheels, J. Terramechanics **4**(2), 7–25 (1967)

50.40 I.C. Schmid: Interaction of vehicle and terrain results from 10 years research at IKK, J. Terramechanics **32**(1), 3–25 (1995)

50.41 L. Ding, Z. Deng, H. Gao, K. Nagatani, K. Yoshida: Planetary rovers' wheel-soil interaction mechanics: New challenges and applications for wheeled mobile robots, Intell. Serv. Robotics **4**(1), 17–38 (2010)

50.42 H. Nakashima, H. Fujii, A. Oida, M. Momozu, Y. Kawase, H. Kanamori, S. Aoki, T. Yokoyama: Parametric analysis of lugged wheel performance for a lunar microrover by means of DEM, J. Terramechanics **44**, 153–162 (2007)

50.43 H. Nakashima, H. Fujii, A. Oida, M. Momozu, H. Kanamori, S. Aoki, T. Yokoyama, H. Shimizu, J. Miyasaka, K. Ohdoi: Discrete element method analysis of single wheel performance for a small lunar rover on sloped terrain, J. Terramechanics **47**, 307–321 (2010)

50.44 W. Li, Y. Huang, Y. Cui, S. Dong, J. Wang: Trafficability analysis of lunar mare terrain by means of the discrete element method for wheeled rover locomotion, J. Terramechanics **47**, 161–172 (2010)

50.45 K. Iagnemma: *A Laboratory single wheel testbed for studying planetary rover wheel-terrain interaction, Tech. Rep. 01-05-05* (MIT, Cambridge 2005)

50.46 S. Wakabayashi, H. Sato, S. Nishida: Design and mobility evaluation of tracked lunar vehicle, J. Terramechanics **46**(3), 105–114 (2009)

50.47 N. Patel, R. Slade, J. Clemmet: The ExoMars rover locomotion subsystem, J. Terramechanics **47**, 227–242 (2010)

50.48 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil, J. Field Robotics **24**(3), 233–250 (2007)

50.49 R. Lindemann, D. Bickler, B. Harrington, G. Ortiz, C. Voorhees: Mars exploration rover mobility development, IEEE Robotics Autom. Mag. **13**(2), 19–26 (2006)

50.50 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based analysis on slope traversability for a planetary exploration rover, Proc. 25th Int. Symp. Space Technol. Sci. (2006) pp. 1025–1030

50.51 S. Michaud, L. Richter, T. Thueer, A. Gibbesch, T. Huelsing, N. Schmitz, S. Weiss, A. Krebs, N. Patel, L. Joudrier, R. Siegwart, B. Schäfer, A. Ellery: Rover chassis evaluation and design optimisation using the RCET, Proc. 9th ESA Workshop Adv. Space Technol. Robotics Autom. (ASTRA) (2006)

50.52 K. Nagatani, A. Ikeda, K. Sato, K. Yoshida: Accurate estimation of drawbar pull of wheeled mobile robots traversing sandy terrain using built-in force sensor array wheel, Proc. 2009 IEEE/RSJ Int. Conf. Robots Syst. (IROS), St. Loius (2009) pp. 2373–2378

50.53 G. Meirion-Griffith, M. Spenko: A Modified pressure-sinkage model for small, rigid wheels on deformable terrains, J. Terramechanics **48**(2), 149–155 (2011)

50.54 C. Senatore, K. Iagnemma: Direct shear behaviour of dry, granular soils for low normal stress with

Avidbots Ex1016
Page 1308 of 2154

application to lightweight robotic vehicle modeling, 17th Conf. Terrain-Veh. Syst. (ISTVS), Blacksburg (2011)
50.55 K. Iagnemma, S. Kang, H. Shibly, S. Dubowsky: Online terrain parameter estimation for wheeled mobile robots with application to planetary rovers, IEEE Trans. Robotics **20**(5), 921–927 (2004)
50.56 S. Hutangkabodee, Y. Zweiri, L. Seneviratne, K. Althoefer: Soil parameter identification for wheel-terrain interaction dynamics and traversability prediction, Int. J. Autom. Comput. **3**(3), 244–251 (2006)
50.57 D. Helmick, A. Angelova, L. Matthies, C. Brooks, I. Halatci, S. Dubowsky, K. Iagnemma: Experimental results from a terrain adaptive navigation system for planetary rovers, Proc. 9th Int. Symp. Artif. Intell., Robotics Autom. Space (i-SAIRAS), Hollywood (2008)
50.58 G. Ishigami, G. Kewlani, K. Iagnemma: A statistical approach to mobility prediction for planetary surface exploration rovers in uncertain terrain, IEEE Robotics Autom. Mag. **16**(4), 61–70 (2009)
50.59 O. Yoshito, K. Nagatani, K. Yoshida, S. Tadokoro, T. Yoshida, E. Koyanagi: Shared autonomy system for traversing and turning tracked vehicles on rough terrain based on continuous three-dimensional terrain scanning, J. Field Robotics **28**(6), 875–893 (2011)

Avidbots Ex1016
Page 1309 of 2154

Multimedia Contents

51. Modeling and Control of Underwater Robots

Gianluca Antonelli, Thor I. Fossen, Dana R. Yoerger

This chapter deals with modeling and control of underwater robots. First, a brief introduction showing the constantly expanding role of marine robotics in oceanic engineering is given; this section also contains some historical backgrounds. Most of the following sections strongly overlap with the corresponding chapters presented in this handbook; hence, to avoid useless repetitions, only those aspects peculiar to the underwater environment are discussed, assuming that the reader is already familiar with concepts such as fault detection systems when discussing the corresponding underwater implementation. The modeling section is presented by focusing on a coefficient-based approach capturing the most relevant underwater dynamic effects. Two sections dealing with the description of the sensor and the actuating systems are then given. Autonomous underwater vehicles require the implementation of mission control system as well as guidance and control algorithms. Underwater localization is also discussed. Underwater manipulation is then briefly approached. Fault detection and fault tolerance, together with the coordination control of multiple underwater vehicles, conclude the theoretical part of the chapter. Two final sections, reporting some successful applications and discussing future perspectives, conclude the chapter. The reader is referred to Chap. 25 for the design issues.

51.1 **The Expanding Role of Marine Robotics in Oceanic Engineering** 1285
51.1.1 Historical Background 1287
51.2 **Underwater Robotics** 1287
51.2.1 Modeling 1287
51.2.2 Sensor Systems 1293
51.2.3 Actuating Systems 1295
51.2.4 Communication Systems 1296
51.2.5 Mission Control System 1296
51.2.6 Guidance and Control 1297
51.2.7 Localization 1299
51.2.8 Underwater Manipulation 1299
51.2.9 Fault Detection/Tolerance 1300
51.2.10 Multiple Underwater Vehicles 1301
51.3 **Applications** 1302
51.4 **Conclusions and Further Reading** 1303
Video-References 1304
References 1304

The world's oceans cover 2/3 of the Earth's surface and have been critical to human welfare throughout history. As in ancient times, they enable the transport of goods between nations. Presently, the seas represent critical sources of food and other resources such as oil and gas. In the near term, we may soon see the emergence of offshore mining for metals as well as the exploitation of gas hydrates. Conversely, the ocean can also threaten human safety and damage infrastructure through natural phenomena such as hurricanes and tsunamis.

51.1 The Expanding Role of Marine Robotics in Oceanic Engineering

Our scientific understanding of the deep sea is expanding rapidly through the use of a variety of technologies. The first scientific explorations were conducted primarily through the use of diving and human-occupied

submersibles, complemented by a variety of other technologies such as towed or lowered instruments, trawls, dredges, autonomous seafloor instruments, and deep-sea drilling. More recently remotely operated and autonomous vehicles have begun to revolutionize seafloor exploration, often returning superior data at reduced costs. In the near future, seafloor observatories linked by fiber-optic cables and satellites will return massive amounts of data from coastal and deep-sea sites. These observations will complement those from conventional expeditionary investigations, and will require teleoperated or robotic intervention during installation and for service. An example of a remotely operated vehicle developed for the scientific study of the seafloor is the Jason 2 vehicle developed at the Woods Hole Oceanographic Institution (shown in Fig. 51.1), and a list of remotely operated vehicles for scientific exploration appears in Table 51.1 (the last vehicle in the table, Kaiko, was lost several years ago).

Offshore oil and gas installations are presently serviced almost exclusively by remotely operated vehicles (ROVs), physically connected via a tether to receive power and data, with human divers used only for the shallowest installations. Subsea systems require extensive work capability during installation, and need frequent inspection and intervention to support drilling operations, actuate valves, repair or replace subsea components, and to accomplish a variety of tasks required to maintain production rates and product quality. The trend toward robotic and teleoperated subsea intervention is certain to continue as offshore oil and gas production moves into deeper waters, and economic considerations push key production steps from surface platforms to the seafloor. Remotely operated manipulators enable these systems to perform complex tasks such as debris removal, cleaning using abrasive tools, and to operate a variety of nondestructive testing tools. The effectiveness of using ROVs decreases with depth mainly due to the cost increase and the difficulties of handling the long tether.



Fig. 51.1 The ROV Jason 2 (courtesy of Woods Hole Oceanographic Institution)

Table 51.1 ROVs for scientific use

Vehicle	Depth (m)	Institution	Manufacturer
Hyperdolphin	3000	JAMSTEC[a]	ISE
Dolphin 3K	3000	JAMSTEC	JAMSTEC
Quest	4000	MARUM[b]	Shilling
Tiburon	4000	MBARI[c]	MBARI
ROPOS	5000	CSSF[d]	ISE
Victor	6000	IFREMER[e]	IFREMER
Jason	6500	WHOI[f]	WHOI
ISIS	6500	NOC[g]	WHOI
UROV 7K	7000	JAMSTEC	JAMSTEC
Kaiko	11 000	JAMSTEC	JAMSTEC

[a] Japan Marine Science and Technology Center; [b] Zentrum für Marine Umweltwissenschaften; [c] Monterey Bay Aquarium Research Institute; [d] Canadian Scientific Submersile Facility; [e] Institut français de recherche pour l'exploitation de la mer; [f] Woods Hole Oceanographic Institution; [g] National Oceanography Centre

Autonomous underwater vehicles (AUVs) are free-swimming unoccupied underwater vehicles that can overcome the limitations imposed by ROV tethers for some tasks. Such vehicles carry their own energy supplies (presently batteries, perhaps fuel cells in the future) and communicate only through acoustics and perhaps optical links in the near future. Limited communications require these vehicles to operate independently of continuous human control, in many cases the vehicles operate completely autonomously. AUVs are currently used for scientific survey tasks, oceanographic sampling, underwater archeology and under-ice survey. Military applications, such as mine detection and landing site survey, are presently operational, and more ambitious applications such as long-term undersea surveillance are in engineering development. Presently, AUVs are incapable of sampling or manipulations tasks like those done routinely by ROVs, as typical work environments tend to be complex and challenging even to skilled human pilots.

Today, approximately 1000 AUVs are operational, many of them experimental. However, they are maturing rapidly. Recently several companies now offer commercial services with AUVs. As an example, for the oil and gas industry the cost reduction of a survey performed with an AUVs instead of a towed vehicle is up to 30% and the data quality is generally higher. Likewise, commercial manufacturers in several countries now offer turnkey AUV systems for specific, well-defined tasks. Currently, remotely operated manip-

Avidbots Ex1016
Page 1311 of 2154

ulators are standard equipment for most ROVs, while on the contrary autonomous manipulation is still a research challenge; the projects SAUVIM [51.1], ALIVE [51.2] and TRIDENT [51.3] were devoted to studying this control problem. VIDEO 88 shows a graphical rendering of a mission with the AUV Nereus, VIDEO 87 shows the perspective view from the Remus AUV, VIDEO 92 a 10-minutes long documentary of an underice mission and finally VIDEO 90 a strong interaction with a shark.

51.1.1 Historical Background

Boats have been used by humans since the start of recorded history, but vehicles able to go under water are more recent. Perhaps the first recorded idea of an underwater machine came from Aristotle; according to legend he built the: *skaphe andros* (boat-man) that allowed Alexander the Great (Alexander III of Macedonia, 356–323 BC) to stay submerged for at least half a day during the war of Tiro in 325 BC. This is probably unrealistic; if true it would precede Archimedes' law, which was first articulated in approximately 250 BC. Leonardo Da Vinci may have been the first to design an underwater vehicle. His efforts were recorded in the *Codice Atlantico* (*Codex Atlanticus*), written between 1480 and 1518. Legends say that Leonardo worked on the idea of an underwater military machine but he destroyed the results as he judged them to be too dangerous. The first use of feedback theory for marine control was probably the Northseeking device, patented in 1908, that used gyroscopic principals to develop the first autopilot [51.4]. From that point on, the use of feedback theory in marine control grew continuously; it is interesting to notice that the proportional–integral–derivative (PID) control commonly used today in numerous industrial applications was first formally analyzed in 1929 by Minorsky [51.5]. The first remotely operated underwater vehicle, POODLE, was built in 1953, and the ROV evolved through the 1960s and 1970s, mostly for military purposes. In the 1980s ROVs became established for use in the commercial offshore industry and began to emerge for scientific applications. The first tetherless, autonomous vehicles were built for experimental purposes in the 1970s. Currently, AUVs are becoming increasingly commonplace for scientific, military, and commercial applications. Turnkey AUV systems for a range of tasks are available from commercial vendors, and AUV services can be acquired from a number of companies [51.6].

51.2 Underwater Robotics

51.2.1 Modeling

A rigid body is completely described by its position and orientation with respect to a reference frame $\Sigma_i, O_i - xyz$ that is supposed to be Earth-fixed and inertial. Let us define $\boldsymbol{\eta}_1 \in \mathbb{R}^3$ as

$$\boldsymbol{\eta}_1 = (x \;\; y \;\; z)^{\mathrm{T}} ,$$

the vector of the body position coordinates in an Earth-fixed reference frame. The vector $\dot{\boldsymbol{\eta}}_1$ is the corresponding time derivative (expressed in the Earth-fixed frame). If one defines

$$\boldsymbol{\nu}_1 = (u \;\; v \;\; w)^{\mathrm{T}}$$

as the linear velocity of the origin of the body-fixed frame $\Sigma_\mathrm{b}, O_\mathrm{b} - x_\mathrm{b} \; y_\mathrm{b} \; z_\mathrm{b}$ with respect to the origin of the Earth-fixed frame expressed in the body-fixed frame (from now on: body-fixed linear velocity) the following relation between the defined linear velocities holds

$$\boldsymbol{\nu}_1 = \mathbf{R}_\mathrm{I}^\mathrm{B} \dot{\boldsymbol{\eta}}_1 , \tag{51.1}$$

where $\mathbf{R}_\mathrm{I}^\mathrm{B}$ is the rotation matrix expressing the transformation from the inertial frame to the body-fixed frame.

Let us define $\boldsymbol{\eta}_2 \in \mathrm{SO}(3)$ as

$$\boldsymbol{\eta}_2 = (\phi \;\; \theta \;\; \psi)^{\mathrm{T}}$$

the vector of body Euler angle coordinates in a Earth-fixed reference frame. In the nautical field those are commonly named roll, pitch, and yaw. Yaw is defined as rotation around the z axis of the fixed frame; pitch is defined as rotation around the y axis resulting after the yaw movement; and roll is defined as rotation around the x axis resulting after both yaw and pitch movements. The vector $\dot{\boldsymbol{\eta}}_2$ is the corresponding time derivative (expressed in the inertial frame). Let us define

$$\boldsymbol{\nu}_2 = (p \;\; q \;\; r)^{\mathrm{T}}$$

as the angular velocity of the body-fixed frame with respect to the Earth-fixed frame expressed in the body-fixed frame (from now on: body-fixed angular velocity). The vector $\dot{\boldsymbol{\eta}}_2$ does not have a physical interpretation and it is related to the body-fixed angular velocity by a proper Jacobian matrix

$$\boldsymbol{\nu}_2 = \mathbf{J}_{k,o}(\boldsymbol{\eta}_2)\dot{\boldsymbol{\eta}}_2 . \tag{51.2}$$

The matrix $\mathbf{J}_{k,o} \in \mathbb{R}^{3\times3}$ can be expressed in terms of the Euler angles as

$$\mathbf{J}_{k,o}(\boldsymbol{\eta}_2) = \begin{pmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{pmatrix}, \tag{51.3}$$

where c_α and s_α are abbreviations for $\cos(\alpha)$ and $\sin(\alpha)$, respectively. The matrix $\mathbf{J}_{k,o}(\boldsymbol{\eta}_2)$ is not invertible for every value of $\boldsymbol{\eta}_2$. In detail, it is

$$\mathbf{J}_{k,o}^{-1}(\boldsymbol{\eta}_2) = \frac{1}{c_\theta} \begin{pmatrix} 1 & s_\phi s_\theta & c_\phi s_\theta \\ 0 & c_\phi c_\theta & -c_\theta s_\phi \\ 0 & s_\phi & c_\phi \end{pmatrix}, \tag{51.4}$$

that it is singular for $\theta = (2l+1)\frac{\pi}{2}$ rad, with $l \in \mathbb{N}$, i. e., for a pitch angle of $\pm\frac{\pi}{2}$ rad.

The rotation matrix $\mathbf{R}_\mathrm{I}^\mathrm{B}$, needed in (51.1) to transform the linear velocities, is expressed in terms of the Euler angles by

$$\mathbf{R}_\mathrm{I}^\mathrm{B}(\boldsymbol{\eta}_2) = \begin{pmatrix} c_\psi c_\theta & s_\psi c_\theta & -s_\theta \\ -s_\psi c_\phi + c_\psi s_\theta s_\phi & c_\psi c_\phi + s_\psi s_\theta s_\phi & s_\phi c_\theta \\ s_\psi s_\phi + c_\psi s_\theta c_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi & c_\phi c_\theta \end{pmatrix}. \tag{51.5}$$

Table 51.2 shows the common notation used for marine vehicles according to the Society of Naval Architects and Marine Engineers (SNAMEs) notation [51.7]; a sketch is shown in Fig. 51.2.

As for any representation of a rigid body's orientation several possibilities arise, among them, the use of a four-parameter description given by unit quaternions. The term *quaternion* was introduced by Hamilton in 1840, 70 years after the introduction of a four-parameter rigid-body attitude representation by Euler. An introduction to alternative orientation representations can be found in Chap. 2 and concerning the marine environment in [51.8].

Table 51.2 Common notation for the motion of a marine vehicle

		Forces and moments	$\boldsymbol{\nu}_1, \boldsymbol{\nu}_2$	$\boldsymbol{\eta}_1, \boldsymbol{\eta}_2$
Motion in the x-direction	Surge	X	u	x
Motion in the y-direction	Sway	Y	v	y
Motion in the z-direction	Heave	Z	w	z
Rotation about the x-axis	Roll	K	p	ϕ
Rotation about the y-axis	Pitch	M	q	θ
Rotation about the z-axis	Yaw	N	r	ψ

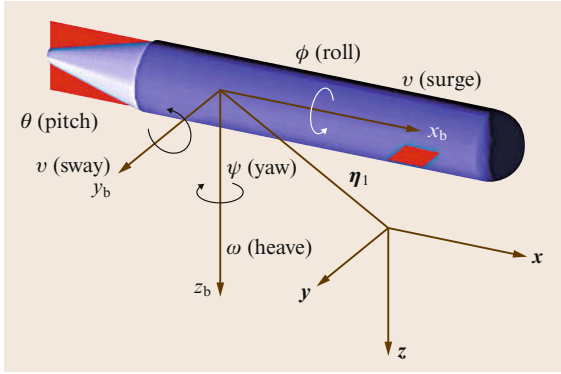


Fig. 51.2 Motion variables for an underwater vehicle

It is useful to collect the kinematic equations in six-dimensional matrix forms. Let us define the vector $\boldsymbol{\eta} \in \mathbb{R}^6$ as

$$\boldsymbol{\eta} = \begin{pmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{pmatrix}, \tag{51.6}$$

and the vector $\boldsymbol{\nu} \in \mathbb{R}^6$ as

$$\boldsymbol{\nu} = \begin{pmatrix} \boldsymbol{\nu}_1 \\ \boldsymbol{\nu}_2 \end{pmatrix}, \tag{51.7}$$

and, defining the matrix $\mathbf{J}_\mathrm{e}(\mathbf{R}_\mathrm{B}^\mathrm{I}) \in \mathbb{R}^{6\times6}$

$$\mathbf{J}_\mathrm{e}(\mathbf{R}_\mathrm{B}^\mathrm{I}) = \begin{pmatrix} \mathbf{R}_\mathrm{I}^\mathrm{B} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{J}_{k,o} \end{pmatrix}, \tag{51.8}$$

where the rotation matrix $\mathbf{R}_\mathrm{I}^\mathrm{B}$ is given in (51.5) and $\mathbf{J}_{k,o}$ is given in (51.3),

$$\boldsymbol{\nu} = \mathbf{J}_\mathrm{e}(\mathbf{R}_\mathrm{B}^\mathrm{I})\dot{\boldsymbol{\eta}}\,. \tag{51.9}$$

The inverse mapping, given the block-diagonal structure of $\mathbf{J}_\mathrm{e}$, is given by

$$\dot{\boldsymbol{\eta}} = \mathbf{J}_\mathrm{e}^{-1}(\mathbf{R}_\mathrm{B}^\mathrm{I})\boldsymbol{\nu} = \begin{pmatrix} \mathbf{R}_\mathrm{B}^\mathrm{I} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{J}_{k,o}^{-1} \end{pmatrix} \boldsymbol{\nu}\,, \tag{51.10}$$

where $\mathbf{J}_{k,o}^{-1}$ is given in (51.4).

Defining as

$$\boldsymbol{\tau}_\mathrm{v} = \begin{pmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{pmatrix},$$

the vector of generalized forces, where

$$\boldsymbol{\tau}_1 = (X \;\; Y \;\; Z)^\mathrm{T}\,, \tag{51.11}$$

Avidbots Ex1016
Page 1313 of 2154

the resultant forces acting on the rigid body expressed in a body-fixed frame, and

$$\boldsymbol{\tau}_2 = (K \;\; M \;\; N)^{\mathrm{T}} , \tag{51.12}$$

the corresponding resultant moment to the pole O_b, it is possible to rewrite the Newton–Euler equations of motion of a rigid body moving in the space as

$$\mathbf{M}_{\mathrm{RB}}\dot{\boldsymbol{\nu}} + \mathbf{C}_{\mathrm{RB}}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_\mathrm{v} . \tag{51.13}$$

The derivation of (51.13) can be found in Chap. 3.

The matrix $\mathbf{M}_{\mathrm{RB}}$ is constant, symmetric, and positive definite, i. e., $\dot{\mathbf{M}}_{\mathrm{RB}} = \mathbf{0}$ and $\mathbf{M}_{\mathrm{RB}} = \mathbf{M}_{\mathrm{RB}}^{\mathrm{T}} > \mathbf{0}$. Its unique parametrization is of the form

$$\mathbf{M}_{\mathrm{RB}} = \begin{pmatrix} m\mathbf{I}_3 & -m\mathbf{S}(\boldsymbol{r}_C^\mathrm{b}) \\ m\mathbf{S}(\boldsymbol{r}_C^\mathrm{b}) & \mathbf{I}_{O_\mathrm{b}} \end{pmatrix} , \tag{51.14}$$

where $\boldsymbol{r}_C^\mathrm{b}$ is the 3×1 distance vector to the center of gravity (CG) expressed in the body-fixed frame, $\mathbf{I}_3$ is the 3×3 identity matrix, and $\mathbf{I}_{O_\mathrm{b}}$ is the inertia tensor expressed in the body-fixed frame $\mathbf{S}(\boldsymbol{x})$ is the matrix operator performing the cross product between two (3×1) vectors

$$\mathbf{S}(\boldsymbol{x}) = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix} .$$

On the other hand, there does not exist a unique parametrization of the matrix $\mathbf{C}_{\mathrm{RB}}$, which represents the Coriolis and centripetal terms. It can be demonstrated that the matrix $\mathbf{C}_{\mathrm{RB}}$ can always be parameterized such that it is skew symmetric, i. e.,

$$\mathbf{C}_{\mathrm{RB}}(\boldsymbol{\nu}) = -\mathbf{C}_{\mathrm{RB}}^{\mathrm{T}}(\boldsymbol{\nu}) \quad \forall \boldsymbol{\nu} \in \mathbb{R}^6 ; \tag{51.15}$$

explicit expressions for $\mathbf{C}_{\mathrm{RB}}$ can be found in [51.8].

Notice that (51.13) can be greatly simplified if the origin of the body-fixed frame is chosen to be coincident with the central frame, i. e., $\boldsymbol{r}_C^\mathrm{b} = \mathbf{0}$.

Hydrodynamic Generalized Forces

Equation (51.13) represents the motion of a rigid body in an empty space, while dealing with ships or underwater vehicles requires the consideration of the presence of the hydrodynamics generalized forces, i. e., the forces and moments caused by the presence of the fluid. In hydrodynamics it is common to assume that the generalized hydrodynamics forces on a rigid body can be linearly superimposed [51.9]; in particular, those are separated into radiation-induced forces, environmental disturbances, and restoring forces due to gravity and buoyancy.

Radiation-induced forces are defined as the:

forces on the body when the body is forced to oscillate with the wave excitation frequency and there are no incident waves;

these can be identified as the sum of the *added mass* due to the inertia of the surrounding fluid and the *radiation-induced potential damping* due to the energy dissipated by generated surface waves.

Environmental disturbances can be identified as the generalized forces caused by the wind, the waves, and the ocean current.

The overall equations of motions can therefore be written in matrix form as [51.8, 10, 11]

$$\mathbf{M}_\mathrm{v}\dot{\boldsymbol{\nu}} + \mathbf{C}_\mathrm{v}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}_\mathrm{v}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}_\mathrm{v}(\mathbf{R}_\mathrm{B}^\mathrm{I}) = \boldsymbol{\tau}_\mathrm{v} , \tag{51.16}$$

where $\mathbf{M}_\mathrm{v} = \mathbf{M}_{\mathrm{RB}} + \mathbf{M}_\mathrm{A}$ and $\mathbf{C}_\mathrm{v} = \mathbf{C}_{\mathrm{RB}} + \mathbf{C}_\mathrm{A}$ also include the added mass terms.

In the following sections these generalized forces, specific to the marine environment, will be briefly discussed.

Added Mass and Inertia

When a rigid body is moving in a fluid, the additional inertia of the fluid surrounding the body that is accelerated by the movement of the body has to be considered. This effect can be neglected in industrial robotics since the density of the air is much lower than the density of a moving mechanical system. In underwater applications, however, the density of the water, $\rho \approx 1000\,\mathrm{kg/m^3}$, is comparable with the density of the vehicles. In particular, at 0 °C, the density of freshwater is $1002.68\,\mathrm{kg/m^3}$; for sea water with 3.5% salinity it is $\rho = 1028.48\,\mathrm{kg/m^3}$.

The fluid surrounding the body is accelerated with the body, so a force is necessary to achieve this acceleration, while the fluid exerts a reaction force which is equal in magnitude and opposite in direction. This reaction force is the added mass contribution. The added mass is not a quantity of fluid to add to the system such that it has an increased mass. Different properties hold with respect to the 6×6 inertia matrix of a rigid body due to the fact that the added mass is a function of the body's surface geometry.

The hydrodynamic force along $\boldsymbol{x}_\mathrm{b}$ due to the linear acceleration in the $\boldsymbol{x}_\mathrm{b}$-direction is defined as

$$X_\mathrm{A} := -X_{\dot{u}}\dot{u} , \quad \text{where} \quad X_{\dot{u}} := X_{\dot{u}} := \frac{\partial X}{\partial \dot{u}} ,$$

where the symbol ∂ denotes the partial derivative. In the same way it is possible to define all the remaining 35 elements that relate the six force/moment components $(X \;\; Y \;\; Z \;\; K \;\; M \;\; N)^{\mathrm{T}}$ to the six linear/angular accelerations $(\dot{u} \;\; \dot{v} \;\; \dot{w} \;\; \dot{p} \;\; \dot{q} \;\; \dot{r})^{\mathrm{T}}$. These elements can be grouped

into the added mass matrix $\mathbf{M}_\mathrm{A} \in \mathbb{R}^{6\times6}$. Usually, all the elements of the matrix are nonzero.

In general, added mass and potential damping will be frequency dependent and depend on the forward speed. This is also the case for certain viscous damping terms (skin friction, roll damping, etc.). This gives a pseudo differential equation describing the frequency response of the vehicle. Since some of the coefficients depend on the frequency this is not an ordinary differential equation (ODE). The frequency equation, however, can be transformed to the time domain using the concepts described in [51.12] and [51.13], and recently in [51.14]. The time-domain equations will in general include fluid memory effects [51.4]. However, underwater robots are usually modelled using maneuvering theory where the potential coefficients are evaluated at the zero frequency. The resulting equation is an ODE where the added inertia matrix $\mathbf{M}_\mathrm{A}$ is constant, speed independent, and positive definite

$$\mathbf{M}_\mathrm{A} = \mathbf{M}_\mathrm{A}^\mathrm{T} > \mathbf{0}\,, \quad \dot{\mathbf{M}}_\mathrm{A} = \mathbf{0}\,. \tag{51.17}$$

This result is well known from ship hydrodynamics; [51.15] for instance. The matrix $\mathbf{M}_\mathrm{A}$ can be computed using numerical programs such as WAMIT or Matlab, based on the US Air Force Digital Datcom [51.16]; in this case, the zero-frequency result should be used, that is, $\mathbf{M}_\mathrm{A} = \mathbf{A}(0)$ where $\mathbf{A}(\omega)$ is the frequency-dependent added mass matrix. The potential damping matrix will be small compared to the viscous effects and drag/lift terms. Hence, this term can be set to zero for underwater vehicles. If the added mass is computed experimentally, it is common practice to symmetrize the results such that

$$\mathbf{M}_\mathrm{A} = \frac{1}{2}\left(\mathbf{A}_\mathrm{exp} + \mathbf{A}_\mathrm{exp}^\mathrm{T}\right),$$

where $\mathbf{A}_\mathrm{exp}$ denotes the experimentally obtained added mass terms.

If the body is completely submerged in the water and is designed with port/starboard symmetry (xz-plane) as is common for underwater vehicles in six degrees-of-freedom (DOF), the following structure of the matrices $\mathbf{M}_\mathrm{A}$ can be considered

$$\mathbf{M}_\mathrm{A} = -\begin{pmatrix} X_{\dot{u}} & 0 & X_{\dot{w}} & 0 & X_{\dot{q}} & 0 \\ 0 & Y_{\dot{v}} & 0 & Y_{\dot{p}} & 0 & Y_{\dot{r}} \\ Z_{\dot{u}} & 0 & Z_{\dot{w}} & 0 & Z_{\dot{q}} & 0 \\ 0 & K_{\dot{v}} & 0 & K_{\dot{p}} & 0 & K_{\dot{r}} \\ M_{\dot{u}} & 0 & M_{\dot{w}} & 0 & M_{\dot{q}} & 0 \\ 0 & N_{\dot{v}} & 0 & N_{\dot{p}} & 0 & N_{\dot{r}} \end{pmatrix}. \tag{51.18}$$

The added mass coefficients can theoretically be derived by exploiting the geometry of the rigid body or numerically by strip theory [51.17].

In [51.18] the coefficients for the experimental AUV Phoenix of the Naval Postgraduate School (NPS) are reported. These coefficients have been derived experimentally, and the geometry gives a nondiagonal $\mathbf{M}_\mathrm{A}$ matrix. To provide an order of magnitude for the added mass terms, for the vehicle mass of about 5000 kg, the $X_{\dot{u}}$ is approximately −500 kg.

The added mass also makes an *added* Coriolis and centripetal contribution. It can be demonstrated that the matrix expression can always be parameterized such that

$$\mathbf{C}_\mathrm{A}(\boldsymbol{\nu}) = -\mathbf{C}_\mathrm{A}^\mathrm{T}(\boldsymbol{\nu})\,, \quad \forall \boldsymbol{\nu} \in \mathbb{R}^6\,,$$

whose symbolic expressions can be found in [51.4].

Hydrodynamic Damping

The hydrodynamic damping for marine vehicles is mainly caused by:

- Potential damping
- Skin friction
- Wave drift damping
- Vortex shedding damping
- Viscous damping.

The radiation-induced potential damping due to forced body oscillations is commonly known as potential damping; its dynamic contribution is usually negligible with respect to, e.g., the viscous friction for underwater vehicles while it may be significant for surface vessels.

Linear skin friction is due to laminar boundary layers and can affect the low-frequency motion of the vehicle. Together with this effect, at high frequencies it is possible to observe a quadratic, or nonlinear, skin friction phenomenon caused by turbulent boundary layers.

Wave drift damping is the dominant dynamic damping effect in surge motion of surface vessels in high sea. It can be considered as an added resistance for boats advancing in waves; its drift is proportional to the square of the significant wave height. In the sway and yaw directions, however, its dynamic contribution is negligible with respect to the effect of vortex shedding.

A body moving in a fluid causes a separation of the flow; this can still be considered as laminar in the upstream while two antisymmetric vortices can be observed in the downstream. In case that the body is a cylinder moving in a direction normal to its axis, the result is a periodic force normal to both the velocity and the axis. This effect may cause the oscillation of cables and other underwater structures. However, concerning underwater vehicles, this effect is negligible for

ROVs and may be counteracted by designing proper small control surfaces for torpedo-like AUVs.

Vortex shedding is an unsteady flow that takes place at special flow velocities (according to the size and shape of the cylindrical body). In this flow vortices are created at the back of the body, periodically from each side.

The viscosity of the fluid also causes dissipative forces. These are composed of drag and lift forces, the former being parallel to the relative velocity of the vehicle with respect to the water while the latter are normal to it. For a sphere moving in a fluid, the drag force can be modeled as [51.9]

$$F_{\text{drag}} = \frac{1}{2}\rho U^2 S C_{\text{d}}(R_{\text{n}}) \,, \tag{51.19}$$

where ρ is the fluid density, U is the velocity of the sphere, S is the frontal area of the sphere, C_{d} is the nondimensional drag coefficient, and R_{n} is the Reynolds number. For a generic body, S is the projection of the frontal area along the flow direction. The drag force can be considered as the sum of two physical effects: the frictional contribution of the surface whose normal is perpendicular to the flow velocity, and the pressure contribution of the surface whose normal is parallel to the flow velocity. For a hydrofoil moving in a fluid, the lift force can be modeled as [51.9]

$$F_{\text{lift}} = \frac{1}{2}\rho U^2 S C_{\text{l}}(R_{\text{n}}, \alpha) \,, \tag{51.20}$$

where S is now the area, C_{l} is the nondimensional lift coefficient, and α is the angle of attack, i. e., the angle between the relative velocity and the tangent to the surface. For small angles of attack, i. e., $|\alpha| < 10^\circ$, the lift coefficient is approximatively proportional to α and rapidly decays to zero as α increases [51.19].

The drag and lift coefficients are therefore dependent on the Reynolds number, i. e., on the laminar/turbulent fluid motion

$$R_{\text{n}} = \frac{\rho |U| D}{\mu} \,,$$

where D is the characteristic dimension of the body perpendicular to the direction of U and μ is the dynamic viscosity of the fluid. Table 51.3 reports the drag coefficients as a function of the Reynolds number for a cylinder [51.20].

A common simplification considers only linear and quadratic damping terms and group these into a matrix $\mathbf{D}_{\text{v}}$ as in (51.16) such that

$$\mathbf{D}_{\text{v}}(\boldsymbol{\nu}) > \mathbf{0} \,, \quad \forall \boldsymbol{\nu} \in \mathbb{R}^6 \,.$$

Gravity and Buoyancy

When a rigid body is completely or partially submerged in a fluid under the effect of the gravity two more forces have to be considered: the gravitational force and buoyancy. The latter is the only hydrostatic effect, i. e., it is not a function of the relative movement between the body and fluid.

Let us define as

$$\boldsymbol{g}^{\text{I}} = (0 \;\; 0 \;\; 9.81)^{\text{T}} \text{m}/s^2$$

the acceleration of gravity. This effect is not constant but varies with depth, longitude, and latitude; however, this value is usually accurate enough for most applications except for inertial navigation systems.

For a completely submerged body the computation of these dynamic effects is straightforward. The submerged weight of the body is defined as $W = m||\boldsymbol{g}^{\text{I}}||$, while its buoyancy $B = \rho \nabla ||\boldsymbol{g}^{\text{I}}||$, where ∇ is the volume of the body and m is its mass. The gravity force, which acts at the center of mass $\boldsymbol{r}_{\text{C}}^{\text{B}}$, is represented in the body-fixed frame by

$$\boldsymbol{f}_{\text{G}}(\mathbf{R}_{\text{I}}^{\text{B}}) = \mathbf{R}_{\text{I}}^{\text{B}} \begin{pmatrix} 0 \\ 0 \\ W \end{pmatrix} ,$$

while the buoyancy force, acting at the center of buoyancy $\boldsymbol{r}_{\text{B}}^{\text{B}}$, is represented in the body-fixed frame by

$$\boldsymbol{f}_{\text{B}}(\mathbf{R}_{\text{I}}^{\text{B}}) = -\mathbf{R}_{\text{I}}^{\text{B}} \begin{pmatrix} 0 \\ 0 \\ B \end{pmatrix} .$$

The 6×1 vector of force/moment due to gravity and buoyancy in the body-fixed frame, included in the left-hand side of the equations of motion, is represented by

$$\boldsymbol{g}_{\text{v}}(\mathbf{R}_{\text{I}}^{\text{B}}) = -\begin{pmatrix} \boldsymbol{f}_{\text{G}}(\mathbf{R}_{\text{I}}^{\text{B}}) + \boldsymbol{f}_{\text{B}}(\mathbf{R}_{\text{I}}^{\text{B}}) \\ \boldsymbol{r}_{\text{G}}^{\text{B}} \times \boldsymbol{f}_{\text{G}}(\mathbf{R}_{\text{I}}^{\text{B}}) + \boldsymbol{r}_{\text{B}}^{\text{B}} \times \boldsymbol{f}_{\text{B}}(\mathbf{R}_{\text{I}}^{\text{B}}) \end{pmatrix} .$$

Table 51.3 Lift and drag coefficient for a cylinder

Reynolds number	Regime motion	C_{d}	C_{l}
$R_{\text{n}} < 2 \times 10^5$	Subcritical flow	1	3–0.6
$2 \times 10^5 < R_{\text{n}} < 5 \times 10^5$	Critical flow	1–0.4	0.6
$5 \times 10^5 < R_{\text{n}} < 3 \times 10^5$	Transcritical flow	0.4	0.6

Avidbots Ex1016
Page 1316 of 2154

In the following, the symbol $\boldsymbol{r}_{\mathrm{G}}^{\mathrm{B}} = (x_{\mathrm{G}}\ y_{\mathrm{G}}\ z_{\mathrm{G}})^{\mathrm{T}}$ (with $\boldsymbol{r}_{\mathrm{G}}^{\mathrm{B}} = \boldsymbol{r}_{\mathrm{C}}^{\mathrm{B}}$) will be used for the center of gravity. The expression for $\boldsymbol{g}_{\mathrm{v}}$ in terms of the Euler angles is

$$\boldsymbol{g}_{\mathrm{v}}(\boldsymbol{\eta}_2) = \begin{pmatrix} (W-B)s_\theta \\ -(W-B)c_\theta s_\phi \\ -(W-B)c_\theta c_\phi \\ -(y_{\mathrm{G}}W - y_{\mathrm{B}}B)c_\theta c_\phi + (z_{\mathrm{G}}W - z_{\mathrm{B}}B)c_\theta s_\phi \\ (z_{\mathrm{G}}W - z_{\mathrm{B}}B)s_\theta + (x_{\mathrm{G}}W - x_{\mathrm{B}}B)c_\theta c_\phi \\ -(x_{\mathrm{G}}W - x_{\mathrm{B}}B)c_\theta s_\phi - (y_{\mathrm{G}}W - y_{\mathrm{B}}B)s_\theta \end{pmatrix} . \tag{51.21}$$

Current

Ocean currents are mainly caused by tidal movement, the atmospheric wind system over the sea surface, heat exchange at the sea surface, salinity changes, the Coriolis force due to the Earth's rotation, nonlinear waves, the major ocean circulations such as the Gulf Stream, the effect of setup phenomena or storm surges, and strong density gradients in the upper ocean. Currents can be very different due to local climatic and/or geographic characteristics; as an example, in fjords, the tidal effect can cause currents of up to 3 m/s, moreover, specific mathematical models exist for the various components [51.8].

Let us assume that the ocean current, expressed in the inertial frame, $\boldsymbol{\nu}_{\mathrm{c}}^{\mathrm{I}}$, is constant and irrotational, i. e.,

$$\boldsymbol{\nu}_{\mathrm{c}}^{\mathrm{I}} = (\nu_{\mathrm{c},x}\ \nu_{\mathrm{c},y}\ \nu_{\mathrm{c},z}\ 0\ 0\ 0)^{\mathrm{T}} ,$$

and $\dot{\boldsymbol{\nu}}_{\mathrm{c}}^{\mathrm{I}} = \boldsymbol{0}$; its effects can be added to the dynamics of a rigid body moving in a fluid simply by considering the *relative* velocity in the body-fixed frame

$$\boldsymbol{\nu}_{\mathrm{r}} = \boldsymbol{\nu} - \mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\boldsymbol{\nu}_{\mathrm{c}}^{\mathrm{I}} \tag{51.22}$$

in the derivation of the added Coriolis and centripetal and the damping terms.

$$\mathbf{M}_{\mathrm{v}}\dot{\boldsymbol{\nu}} + \mathbf{C}_{\mathrm{RB}}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{C}_{\mathrm{A}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} + \mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} + \boldsymbol{g}_{\mathrm{v}}(\mathbf{R}_{\mathrm{B}}^{\mathrm{I}}) = \boldsymbol{\tau}_{\mathrm{v}} . \tag{51.23}$$

Notice that the term $\mathbf{C}_{\mathrm{A}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}}$ includes the important destabilizing effect known as the Munk moment [51.9].

If $\mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}_{\mathrm{r}})$ is unknown, quadratic surge resistance and the cross-flow drag principle can be used to describe the dissipative forces and moments in surge, sway, and yaw [51.9]. Moreover

$$\mathbf{C}_{\mathrm{A}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} + \mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} \approx (X_{\mathrm{c}}\ Y_{\mathrm{c}}\ 0\ 0\ 0\ N_{\mathrm{c}})^{\mathrm{T}} \tag{51.24}$$

for large relative current angles $|\beta_{\mathrm{c}} - \psi|$, where β_{c} is the current direction, the cross-flow principles models the sway force Y_{c} and yaw moment N_{c} as

$$Y_{\mathrm{c}} = \frac{\rho}{2}\int_L H(x)C_{\mathrm{D}}(x)v_{\mathrm{r}}^x(x)|v_{\mathrm{r}}^x(x)|\mathrm{d}x \tag{51.25}$$

$$N_{\mathrm{c}} = \frac{\rho}{2}\int_L xH(x)C_{\mathrm{D}}(x)v_{\mathrm{r}}^x(x)|v_{\mathrm{r}}^x(x)|\mathrm{d}x - X_{r|r|}r|r| , \tag{51.26}$$

where L is the vehicle length, $H(x)$ is the vehicle height, $C_{\mathrm{D}}(x)$ is the two-dimensional drag coefficient, and $v_{\mathrm{r}}^x(x) = v_{\mathrm{r}} + rx$ is the relative cross-flow velocity at x. In practice $C_{\mathrm{D}}(x)$ can be chosen as a constant between 0 and 1. The proper value can be determined by curve fitting of experimental data. Along the surge direction, however, the quadratic damping contribution X_{c} is still well represented by a term proportional to the square of the relative velocity, whose symbolic expression can be written as

$$X_{\mathrm{c}} = -X_{u|u|}u_{\mathrm{r}}|u_{\mathrm{r}}| , \tag{51.27}$$

where $-X_{u|u|} > 0$ is the quadratic surge damping coefficient, which can be found by curve fitting of experimental data or relating it to the drag coefficient C_{d} as in (51.19). The approximation embedded in (51.24) does not represent sufficiently enough the dynamics at low velocities, when the quadratic terms are negligible. Hence, it is common to add a linear optional damper in surge, sway and yaw; this can be physically explained as skin friction which shows a linear behavior.

Alternatively, the computation of the quadratic surge resistance, nonlinear roll damping, and the cross-flow drag effect can be made by resorting to the Datcom database for aircraft, as shown in [51.21].

Model Properties

For completely submerged bodies in an ideal fluid moving at low velocity where there are no currents or waves, (51.16) satisfies the following properties:

- The inertia matrix is symmetric and positive definite, i. e.,
$$\mathbf{M}_{\mathrm{v}} = \mathbf{M}_{\mathrm{v}}^{\mathrm{T}} > \mathbf{0} .$$
- The damping matrix is positive definite, i. e.,
$$\mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}) > \mathbf{0} .$$
- The matrix $\mathbf{C}_{\mathrm{v}}(\boldsymbol{\nu})$ is skew symmetric, i. e.,
$$\mathbf{C}_{\mathrm{v}}(\boldsymbol{\nu}) = -\mathbf{C}_{\mathrm{v}}^{\mathrm{T}}(\boldsymbol{\nu}), \forall \boldsymbol{\nu} \in \mathbb{R}^6 .$$

Hydrodynamic Modeling

The mathematical model of an underwater robot as expressed in (51.16) is of great importance; even when

Table 51.4 Dorado payload for scientific sampling (courtesy of Monterey Bay Aquarium Research Institute [51.22])

Sensor model	Description
WHN300	300 kHz Acoustic Doppler Current Profiler/Doppler Velocity Log, manufactured by Teledyne/RD Instruments
8CB4000I	Paroscientific Digiquartz pressure sensor, 4000 m full scale
Gulpers	10 × 2 liter Water Samplers
HS2	Hobilabs 2 channel backscatter/fluorometer, 420 nm/700 nm excitation
LISST-100	Sequoia Scientific particle size spectrum instrument
ISUS	In-situ ultraviolet spectrometer for nitrate measurement
2xSBE3/SBE4	Seabird temperature/conductivity instrument, 2 pairs of instruments carried
SBE43	Seabird dissolved oxygen cell
2xSBE5	Pumps for C/T/DO/ISUS and C/T/LISST flow paths
UBAT	Wetlabs bioluminescence assessment tool (bathyphtometer for measuring plankton bio-luminsescence)
LOPC	Laser Optical Plankton Counter
ECO-CDOM	Fluorometer for measuring CDOM (colored dissolved organic matter)

simplified it captures the most important part of the dynamics. Moreover, it is in a form appropriate for control design. A wide literature exists on AUV/ROV controllers whose stability relies on the properties reported above. On the other hand, there are working conditions in which these assumptions are no longer valid, i. e., when the AUV is traveling at high speed, or close to the surface, or when its shape does not allow geometric simplifications. The latter is the case of, e.g., several ROVs. In addition, it is still common to design the controllers for AUVs based on linearized models and to control ROVs with simple PID controllers.

These considerations justify a modeling effort to calculate the hydrodynamic terms more accurately with the aim of prediction, simulation, and performance analysis rather than control design. This can be done by switching from a coefficient-based approach, such as that presented above, to a component modeling method, the latter being based on computational fluid dynamics theory. In detail, each vehicle geometry, with its specific angle of attack and sideslip, is taken into consideration when computing the hydrodynamic forces/moments. This increased computational effort makes it possible to capture some dynamic effects, such as the vortex-induced roll moment, not justifiable with the coefficient-based approach.

The control plant model is usually a simplified model that captures the most important parts of the dynamics. The most accurate model of the vehicle should be used for prediction and motion simulation.

51.2.2 Sensor Systems

Underwater vehicles are equipped with a sensor system devoted to enabling motion control as well as accomplishing the specific mission it has been commanded to complete. In the latter case, sensors developed for chemical/biological measurements or mapping may be installed, which is beyond the scope of this chapter. For seek of example, Table 51.4 reports the payload for the AUV Dorado, shown in Fig. 51.3.

AUVs need to operate underwater most of the time; one of the major problems with underwater robotics is in the localization task due to the absence of a single, proprioceptive sensor to measure the vehicle position. The global positioning system (GPS) cannot be used underwater. Redundant multisensor systems are commonly combined using state estimation or sensor fusion techniques to provide fault detection and tolerance capability to the vehicle. Table 51.5 lists the types of sensors and the corresponding variable measured commonly available for unmanned underwater vehicles (UUVs).

The sensors that can be found on an underwater vehicle are:

- *Compass*: A gyrocompass can provide an estimate of geodetic north accurate to a fraction of a degree. Magnetic compasses can provide estimates of

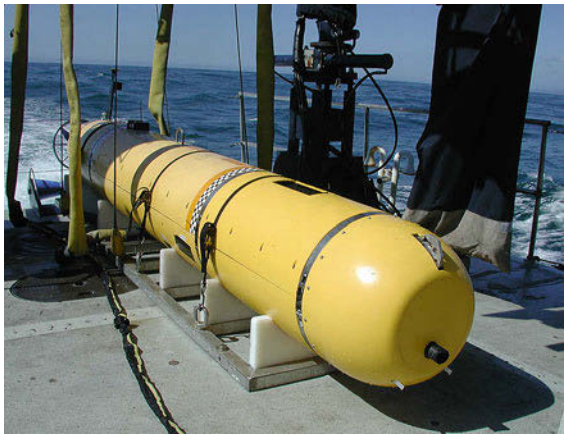


Fig. 51.3 The AUV Dorado (courtesy of Monterey Bay Aquarium Research Institute [51.22])

Avidbots Ex1016
Page 1318 of 2154

Table 51.5 UUVs: possible instrumentation

Sensor	Measured variable
Inertial system	Linear acceleration and angular velocity
Pressure meter	Vehicle depth
Frontal sonar	Distance from obstacles
Vertical sonar	Distance from the bottom
Doppler velocity log	Relative velocity vehicle/bottom
Current meter	Relative velocity vehicle/current
Global positioning system	Absolute position at the surface
Compass	Orientation
Acoustic positioning	Absolute position in known area
Vision systems	Relative position/velocity
Acoustic Doppler current profiler	Relative water velocity

magnetic north with an accuracy of less than 1° if carefully calibrated to compensate for magnetic disturbances from the vehicle itself. Tables or models can be used to convert from magnetic north to geodetic north.

- *Gyroscope*: The term gyroscope, or gyro, is the name given generically to any instrument measuring inertial angular rotation. Those are based on inertial properties of a vibrating mass or of light. Very accurate gyros may output signal that, when integrated, give heading information with a small enough drift for practical purposes.
- *Inertial measurement unit* (*IMU*): An IMU provides information about the vehicle's linear acceleration and angular velocity. These measurements are combined to form estimates of the vehicle's attitude including an estimate of geodetic (true) north from the most complex units. In most cases, for slow-moving underwater vehicles, an independent measurement of the vehicle's velocity is also required to produce accurate estimates of the translational velocity or relative displacement.
- *Depth sensor*: Measuring the water pressure gives the vehicle's depth. At depths beyond a few hundred meters, the equation of state of seawater must be invoked to produce an accurate depth estimate based on the ambient pressure [51.23]. With a high-quality sensor, these estimates are reliable and accurate, giving a small error of order 0.01%.
- *Altitude and forward-looking sonar*: These are used to detect the presence of obstacles and distance from the seafloor.
- *Doppler velocity log* (*DVL*): By processing reflected acoustic energy from the seafloor and the water column from three or more beams, estimates of vehicle velocity relative to the seafloor and relative water motion can be obtained. Bottom-tracking velocity estimates can be accurate to $\approx 1\,\mathrm{mm/s}$.
- *Global navigation satellite system* (*GLS*): This is used to localize the vehicle while on the surface to initialize or reduce drift of estimates from an IMU/DVL combination. GNSS, such as GPS or Galileo, only works at the surface.
- *Acoustic positioning*: A variety of schemes exist for determining vehicle position using acoustics. Long-baseline navigation can determine the position of the vehicle relative to a set of acoustic beacons anchored to the seafloor or on the surface through range estimates obtained from acoustic travel times. Ultrashort-baseline navigation uses phase information to determine direction from a cluster of hydrophones; this is most often used to determine the direction of the vehicle (in two dimensions) from a surface support vessel, which is then combined with an acoustic travel-time measurement to produce an estimate of relative vehicle position in

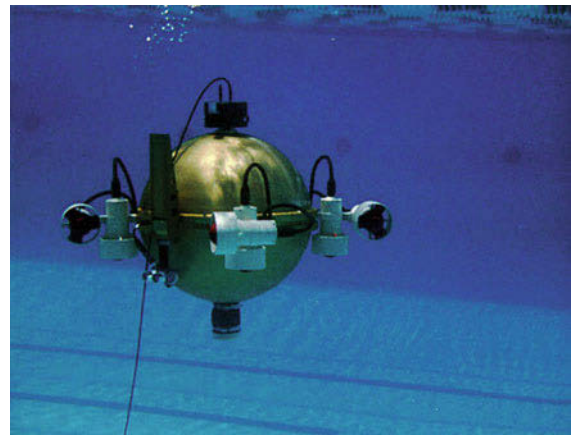


Fig. 51.4 The fully actuated AUV ODIN (courtesy of Autonomous Systems Laboratory, University of Hawaii [51.24])

Table 51.6 JHUROV instrumentation

Measured variable	Sensor	Precision	Update rate
3-DOF vehicle position	SHARP acoustic transponder	0.5 cm	10 Hz
Depth	Foxboro/ICT model n. 15	2.5 cm	20 Hz
Heading	Litton LN200 IMU Gyro	0.01°	20 Hz
Roll and pitch	KVH ADGC	0.1°	10 Hz
Heading	KVH ADGC	1°	10 Hz

Avidbots Ex1016
Page 1319 of 2154

Table 51.7 ODIN III sensors summary

Measured variable	Sensor	Update rate
xy vehicle position	8 sonars	3 Hz
Depth	Pressure sensor	30 Hz
Roll, pitch, and yaw	IMU	30 Hz

spherical coordinates. These techniques will be discussed later in the localization section.

- *Vision systems*: Cameras can be used to obtain estimates of relative, and in some cases absolute, motion using a type of simultaneous localization and mapping (SLAM) algorithm [51.25] and used to perform tasks such as visual tracking of pipelines, station keeping, visual servoing or image mosaicking.

As an example, Table 51.6 reports some data from the instrumentation of the ROV developed at the John Hopkins University [51.26], and Table 51.7 some data from the AUV ODIN III [51.27] shown in Fig. 51.4. *Majumder* et al. [51.28] show some data fusion results with a redundant sensorial system mounted on the AUV Oberon, while [51.29] reviews advances in navigation technology.

51.2.3 Actuating Systems

Marine vehicles are generally propelled by means of thrusters or hydrojets. In the case of ROVs with structural pitch–roll stability, there are usually four thrusters that provide holonomic mobility to the four remained DOFs, in particular, the depth is often decoupled and the vehicle is controlled on a plane in the surge, sway, and yaw DOFs. Those vehicles, being underactuated, cannot easily be used for interactive control by means of a manipulator due to the impossibility of counteracting the generalized forces exchanged with the manipulator's base; in such case, six or more thrusters are required. AUVs generally have a torpedo-like shape and are used for mapping/exploration. They are propelled using one or two thrusters parallel to the fore–aft direction and a fin and a rudder; this kind of propulsion is obviously nonholonomic and experiences a loss of mobility at low velocities. Hydrojets, also known as pump jets or water jets, are systems that create a jet of water for propulsion; they have certain advantages over thrusters such as a higher power density and usability in shallow water, but can provide thrust in one direction only.

Several efforts have been made to accurately and efficiently describe the mathematical model of a thruster; [51.30] reports a *one-state* model where the state is n, the propeller shaft speed. In [51.31] a *two-state* model is proposed to take into account the experimentally observed overshoot in the thrust; together with n, the additional state variable is u_p, the axial flow velocity in the propeller disc. In [51.32] a thruster model incorporating the effects of rotational fluid velocity and inertia on thruster responses is presented together with a method for experimentally determining nonsinusoidal lift/drag curves. A *three-state* model is described in [51.33]

$$\begin{aligned} J_\mathrm{m}\dot{n} + K_n n &= \tau - Q\,, \\ m_f\dot{u}_\mathrm{p} + d_{f0}u_\mathrm{p} + d_f|u_\mathrm{p}|(u_\mathrm{p} - u_\mathrm{a}) &= T\,, \\ (m - X_{\dot{u}})\dot{u} - X_u u - X_{u|u|}u|u| &= (1-t)T\,, \end{aligned}$$

where J_m is the moment of inertia for the dc-motor/propeller, K_n is the linear motor damping coefficient, τ is the motor control input, Q is the propeller torque, m_f is the mass of water in the propeller control volume, u_p is the axial flow velocity in the propeller disc, d_{f0} and d_f are the linear and quadratic damping coefficients for control volume, respectively, u_a is the ambient water velocity, T is the propeller thrust, and t is the thrust deduction number (Fig. 51.5). In the case of steady-state motion, i. e., $\dot{u} = 0$, the ambient water velocity u_a is related to the surge by the *wake fraction number* w as

$$u_\mathrm{a} = (1-w)u\,. \tag{51.28}$$

Notice also that the unmeasured variable u_p can be estimated using a nonlinear observer [51.33].

The *outputs* of the nonlinear three-state dynamic systems are the thrust T and the torque Q, which are functions of several variables; in the following, unsteady flow effects such as air suction, cavitation, the in-and-out-of-water (Wagner), boundary layer, and

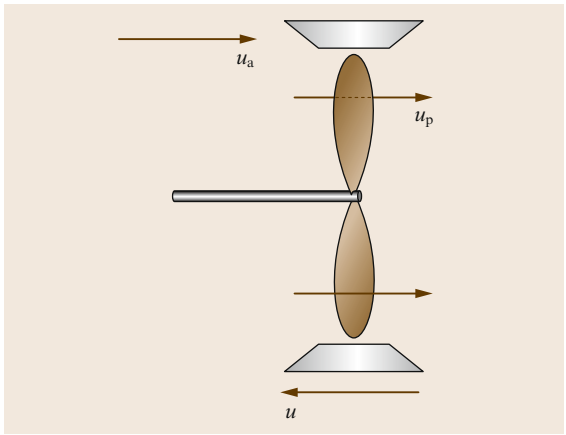


Fig. 51.5 Ambient water and axial flow velocities affecting thruster behavior

Avidbots Ex1016
Page 1320 of 2154

gust (Kuessner) effects will be neglected. This leads to a quasi-steady representation of the model

$$T = \rho D^4 K_T(J_0) n|n| \,, \tag{51.29}$$

$$Q = \rho D^5 K_Q(J_0) n|n| \,, \tag{51.30}$$

where D is the propeller diameter and $K_T(J_0)$ and $K_Q(J_0)$ are the thrust and torque coefficients, respectively. The latter are function of the advance ratio J_0

$$J_0 = \frac{u_a}{nD} \,. \tag{51.31}$$

The open-water propeller efficiency in undisturbed water is given as the ratio of the work done by the propeller in producing a thrust force divided by the work required to overcome the shaft torque

$$\eta_0 = \frac{u_a T}{2\pi n Q} = \frac{J_0}{2\pi} \cdot \frac{K_T}{K_Q} \,. \tag{51.32}$$

Figure 51.6 shows the values of K_T, K_Q, and η_0 as functions of the advance ratio for the Wageningen B4-70 propeller [51.34].

Controlling a marine vehicle usually requires that desired forces/moments act on the vehicle's body; these generalized forces are mapped into desired thrusts to be provided by the propellers. There is, thus, a nontrivial control problem in that the motors are required to provide the appropriate propeller shaft speed n that satisfies the nonlinear relationship with the thrust T presented above.

To enable robustness with respect to possible failures, the actuating system is often redundant. In this case, the problem of allocation of the desired force/moment acting on the vehicle among the thrusters must also be solved. Reference [51.35] reports a survey of control allocation methods of ships and underwater vehicles.

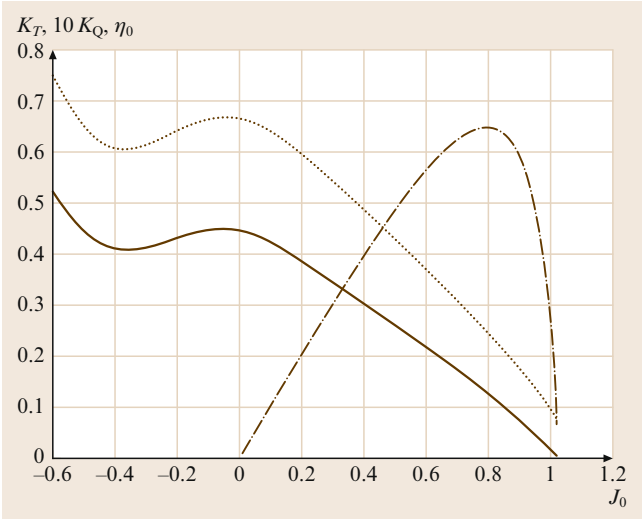


Fig. 51.6 Values of K_T (*solid*), $10K_Q$ (*dotted*), and η_0 (*dash-dotted*) as a function of J_0 (after [51.34])

51.2.4 Communication Systems

Autonomous underwater vehicles may require to communicate with a remote operator or with a base station for monitoring purposes. In case of coordinated mission among several AUVs a network of vehicles/modes communicating among them need to be developed.

Due to the physical characteristics of the water, the more diffused communication technology is based on acoustic propagation. However, the performance of acoustic modems are not as efficient as their aerial conuterparts. Three main factors affect the acoustic propagation: a low speed of sound ($\approx 1500\,\mathrm{m/s}$), the presence of time-varying multiple paths, an attenuation increasing with the frequency. Finally, the channel capacity is function of the distance and is limited [51.36]. For such a challenging communication medium, efficient communication protocols need to be properly designed [51.37].

51.2.5 Mission Control System

The mission control system (MCS) can be considered as the highest-level process running during an AUV's mission; it is responsible for achieving several control objectives. At the highest level it works as an interface between the operator, accepting his instructions in a higher-level language and decomposing those instructions into mission tasks according to the implemented software architecture. The mission tasks are generally concurrent and their handling depends on the vehicle state and environmental conditions; it is therefore the MCS that handles the tasks, eventually suppressing, sequencing, modifying, and prioritizing them. An MCS is also usually equipped with a graphical user interface (GUI) to report the mission state to the operator, see, e.g., the Neptus Command and Control Infrastructure in VIDEO 324.

As for most advanced robotics applications, an efficient MCS should allow the use of complex robotic systems by users that do not necessarily know all of their technical details. An overview relevant to underwater mission control is given in [51.38], which includes an interesting classification of the MCSs in use in several laboratories according to which four major AUV control architectures were identified: the hierarchical, heterarchical, subsumption, and hybrid.

Avidbots Ex1016
Page 1321 of 2154

From a mathematical point of view, the MCS generally needs to be designed in order to be able to address hybrid dynamical systems, i. e., handling both event-driven and time-driven processes. In [51.39], e.g., the MCS developed at the Portuguese Instituto Superior Técnico (IST), named CORAL, is implemented by resorting to a Petri-net-based architecture that properly handles all the necessary tasks in order to manage navigation, guidance and control, sensing, communications, etc.

The motion-oriented operating system (MOOS), designed at the Massachusetts Institute of Technology, is a software tool capable of executing and coordinating a multitude of subsea operations. The MCS developed at the Naval Postgraduate School is in the framework of the behavioral control organized in three layers [51.40]; it is based on PROLOG, an artificial intelligence language for predicate logic.

51.2.6 Guidance and Control

The terms guidance and control can be defined as [51.8]:

- *Guidance* is the action of determining the course, attitude, and speed of the vehicle, relative to some reference frame (usually the Earth), to be followed by the vehicle.
- *Control* is the development and application to a vehicle of appropriate forces and moments for operating point control, tracking, and stabilization. This involves designing the feedforward and feedback control laws.

Figure 51.7 shows the corresponding block diagram, in which the navigation component is also outlined.

Guidance of Underwater Vehicles

Guidance algorithms may benefit from a wide range of inputs, overall mission information, real-time operator input, environmental measured data such as the ocean current, environmental topological information such as a bathymetric map, exteroceptive sensors for obstacle avoidance, and obviously the vehicle state as output from the navigation system.

The vehicle may be required to follow a path, i. e., a curve geometrically represented in two or three dimensions, or a trajectory, i. e., a path with a specific time law assigned. Moreover, when the desired position is constant, the problem is called set-point regulation or maneuvering. The guidance problem is commonly decomposed into simple subtasks of lower dimension: an attitude control problem and a path control. Moreover, attitude is usually considered as a simple depth set-point with null roll and pitch and the path is usually a line in the horizontal plane.

One of the most common guidance approaches is based on the generations of way-points. Those are usually stored in a database and are used to generate the vehicle path/trajectory; a *passing* velocity, in fact, may be defined together with the Cartesian coordinates of the points. The simplest way to connect the way-points is to use the segments connecting two successive way-points. Efficient way-point-based guidance approaches need to take into account the presence of the current and the eventual nonholonomicity of the vehicle [51.41]. A technique for adaptively tracking bathymetric contours by proper generation of way-points is presented in [51.42]; environment information is acquired by mean of a single vertical sonar. An alternative method is based on *line-of-sight* guidance [51.43–45]. In this case, the heading control is computed by considering as input the angle formed by the vector from the vehicle to the next way-point rather than requiring the vehicle to

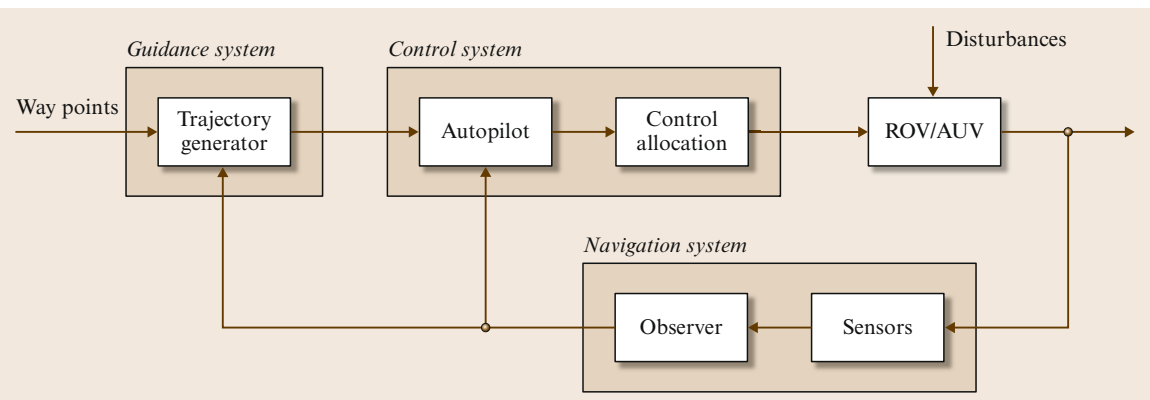


Fig. 51.7 Guidance, navigation, and control for an autonomous marine vehicle

exactly follow the line segment between the current and the following way-point. For docking maneuvers, algorithms must enable control that meets the precision and alignment constraints of the docking fixture [51.46].

By combining vision-based guidance with a neurocontroller trained by reinforcement learning, in [51.47], an algorithm aimed at a hold station on a reef or swimming along a pipe has been presented. In [51.48] guidance for AUVs specifically involved in a predeployment survey of the sea bottom and visual inspection of pipelines is given. *Hyland* and *Taylor* [51.49] report a specific guidance system aimed at mine avoidance for AUVs. Based on a three-dimensional discretization of the environment, the path-planning technique consists of computing a safe path avoiding the unsafe cells of the map. Due to the poor manoeuvrability at low speed under some conditions, the vehicle has to make a 360° turn to avoid stopping and to map the environment close to it before generating a safe path.

A deep discussion on guidance for surface and underwater vehicles can be found in [51.4, 8].

Control of Underwater Vehicles

Control of underwater vehicles needs to consider the different operating conditions and actuating configurations in which a submerged vehicle is required to operate. In particular, there are three main control problems:

- An AUV traveling at high speed ($>1\,\mathrm{m/s}$) generally equipped with at least one thruster aligned in the fore–aft direction and at least two control surfaces (stern and rudder).
- An underactuated ROV, with high metacentric stability, i. e., structurally stable in roll and pitch, and equipped with at least four thrusters.
- A fully actuated AUV equipped with at least six trusters.

AUVs equipped with control surfaces are underactuated vehicles mainly used for survey/exploration missions. Inheriting the common practice of submarine control, they are not allowed to perform arbitrary motions in six DOFs but are rather designed to perform specific movements such as: cruising along a given direction at constant depth, steering at constant depth, or diving. Marine experience and mathematical insight, in fact, demonstrate that these movements are lightly coupled in dynamic terms. For these vehicles, moreover, specific manoeuvres such as homing or docking require special capabilities [51.46]. This requires the design of vehicles that are structurally stable in the roll DOF. Cruising requires control of the surge velocity $u(t)$; steering requires control of the sway velocity $v(t)$ and the yaw DOF $r(t)$, $\psi(t)$, diving requires control of the heave DOF $\omega(t)$, $z(t)$ and the pitch DOF $q(t)$, $\theta(t)$. The simplest configuration of actuators that can control an AUV through these movements is composed of one thruster aligned along the fore–aft direction, one stern, and one rudder; the control variables, thus, are the propeller speed and the deflection of the fins. Several approaches can then be considered to solve this control problem, among them, in [51.50] the sliding mode control is proposed, while [51.51] presents an adaptive sliding mode control for the diving manoeuvre. *Healey* and *Lienard* [51.18] reports a successful implementation of multivariable sliding mode control on the NPS AUV II, later also implemented on the NPS ARIES AUV [51.52]. As the model of an AUV traveling at high speed is nonlinear and coupled, the tuning of the parameters is mainly based on a linearized model around the working conditions.

From a descriptive point of view, an ROV is mainly a box-shaped underwater vehicle equipped with tools such as a video camera or robot manipulator, while its payload is often variable depending on the task. It is remotely operated and physically connected to another vehicle, either an underwater or a surface vessel. It is mainly designed to travel at low speed and it is structurally stable in roll and pitch, while its depth, surge, sway, and yaw are independently controllable. Due to the absence of a specific shape, the varying payload, and the relatively low required performances, it is common to control a ROV by means of single-input single-output (SISO) controllers. Moreover, the PID approach is often used due to its simplicity. A two-layer guidance and control architecture for the ROV Romeo is given in [51.53].

Control of a fully actuated AUV in six DOFs is needed in the case of, e.g., an interaction task performed by a manipulator mounted on a vehicle; the latter, in fact, needs to provide all the force/moment components in order to counteract the presence of the manipulator dynamically. This problem is kinematically similar to the problem of controlling a satellite in six DOFs; the underwater environment, however, makes it significatively different from the dynamic point of view. In kinematic terms the main issue is in implementing a suitable policy for orientation control; any three-parameter representation of orientation, in fact, experiences representation singularities (Chap. 2). This problem may be overcome by resorting to a redundant representation of the orientation such as the quaternion. Most of the six-DOF controllers proposed in the literature are based on (51.16), which model the simplified effect of the hydrodynamic terms and which have very similar properties as the equations of motion of an industrial manipulator. Based on this, it is obviously possible to find a collection of approaches inherited

Avidbots Ex1016
Page 1323 of 2154

from classical robotics, [51.4, 8] for some examples. In [51.54], some specific considerations for the underwater environment lead to a quaternion-based, adaptive controller; it is worth noticing that adaptive control requires a suitable, and simplified, expression for the hydrodynamic terms. In [51.55] a comparison among several 6-DOF controllers is made. VIDEO 267 and VIDEO 268 show two controllers validation implemented in a pool.

51.2.7 Localization

Localization in the underwater environment can be a complex task, mainly due to the absence of a single external sensor that gives the vehicle position such as, e.g., the GPS for outdoor ground vehicles; moreover, the environment is often poorly structured.

One of the most reliable methods is based on the use of acoustic systems such as the baseline systems: the long-baseline system (LBL), the short-baseline system (SBL), and the ultrashort-baseline system (USBL). These systems are based on the presence of a transceiver mounted on the vehicle and a variable number of transponders located in known positions. The transceiver's distance from each transponder can be measured via the measurement of an echo delay; from this information the position of the vehicle can be calculated by basic triangulation operations [51.56]. The USBL can be used with a single transponder, which is usually mounted on a surface ship whose position is measured by GPS. Recent works are aimed at developing localization algorithms based on a single beacon, i. e., range-only measurement among vehicles and buoys [51.57].

Another localization system is called terrain-aided navigation and is based on the use of terrain elevation maps; bathymetric maps are available, especially in the case of well-known locations such as harbors where they usually have a resolution of ≈ 1 m. In this case, the vehicle position is obtained by filtering the information coming from a downward-looking sonar. In [51.58], a particle filter approach was used to localize an AUV in Sydney harbor.

Moving vehicles may be equipped with an IMU or DVL in order to measure its velocity and/or acceleration. This data can then be integrated to estimate the vehicle position. This kind of information is subject to the drift phenomenon and may not be reliable for long-duration runs or may become cost ineffective if accurate IMU devices are needed.

Relative localization can be obtained by resorting to any device that provides information about the relative position of the vehicle with respect to the environment, even in the absence of a map. In this case, by filtering the distance measurements taken along the motion, the vehicle's position can be measured. This is the case, e.g., of sonar or vision-based localization techniques [51.59].

Often, the techniques presented above are used together in a redundant system and the effective position is obtained by resorting to sensor fusion techniques such as the Kalman filtering approach.

Simultaneous Localization and Mapping

Simultaneous localization and mapping (SLAM), also known as concurrent mapping and localization (CML), is a wide topic in mobile robotics. The problem can be formulated as the requirement for a mobile robot to be placed in an unknown environment and progressively build a map while locating itself inside the map. Chapter 46 discusses this topic in detail. For the marine environment an additional issue arises due by the large-scale map that needs to be used for long-duration missions; [51.60] implements a decoupled stochastic mapping to handle this computational problem in an extended Kalman filter. Terrain-aided navigation with the use of a scanning sonar is implemented in [51.61]. *Newman* and *Leonard* [51.62] use long-baseline range measurements as the input for a nonlinear least-squares approach solved by the Gauss–Newton method; both the initially unknown position of the transponders and the vehicle position are estimated. An interesting survey on navigation and SLAM for underwater vehicles is given in [51.29].

51.2.8 Underwater Manipulation

A manipulator may be mounted on an AUV or a ROV in order to accomplish interaction operations. In this case, the vehicle needs to be fully actuated to counteract the forces and moments generated by the manipulator's base. By considering a manipulator with n links, thus six DOFs, the underwater vehicle manipulator system (UVMS) is a $(6+n)$-DOF robotic system whose velocity vector is

$$\boldsymbol{\zeta} = (\boldsymbol{\nu}_1^{\mathrm{T}} \ \boldsymbol{\nu}_2^{\mathrm{T}} \ \dot{\boldsymbol{q}}^{\mathrm{T}})^{\mathrm{T}} , \tag{51.33}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the vector collecting the manipulator joints positions.

Repeating the same considerations as for an underwater vehicle, it is possible to write the equations of motions of an UVMS in matrix form as

$$\mathbf{M}(\boldsymbol{q})\dot{\boldsymbol{\zeta}} + \mathbf{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \mathbf{D}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{g}(\boldsymbol{q}, \mathbf{R}_{\mathrm{B}}^{\mathrm{I}}) = \boldsymbol{\tau} , \tag{51.34}$$

where $\mathbf{M} \in \mathbb{R}^{(6+n)\times(6+n)}$ is the inertia matrix, including added mass terms, $\mathbf{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} \in \mathbb{R}^{6+n}$ is the vector of

Avidbots Ex1016
Page 1324 of 2154

the Coriolis and centripetal terms, $\mathbf{D}(\boldsymbol{q},\boldsymbol{\zeta})\boldsymbol{\zeta} \in \mathbb{R}^{6+n}$ is the vector of dissipative effects, and $\boldsymbol{g}(\boldsymbol{q},\mathbf{R}_1^{\mathrm{B}}) \in \mathbb{R}^{6+n}$ is the vector of gravity and buoyancy effects. The relationship between the generalized forces $\boldsymbol{\tau}$ and the control input is given by

$$\boldsymbol{\tau} = \begin{pmatrix} \boldsymbol{\tau}_{\mathrm{v}} \\ \boldsymbol{\tau}_q \end{pmatrix} = \begin{pmatrix} \mathbf{B}_{\mathrm{v}} & \mathbf{0}_{6\times n} \\ \mathbf{0}_{n\times 6} & \mathbf{I}_n \end{pmatrix} \boldsymbol{u} = \mathbf{B}\boldsymbol{u}\,, \qquad (51.35)$$

where $\boldsymbol{u} \in \mathbb{R}^{p_{\mathrm{v}}+n}$ is the vector of the control input. Notice that, while for the vehicle a generic number $p_{\mathrm{v}} \geq 6$ of control inputs is assumed, for the manipulator it is supposed that n joint motors are available.

Under this hypothesis, which can be considered as reasonable at low velocity, it holds that:

- The inertia matrix $\mathbf{M}$ of the system is symmetric and positive definite.
- For a suitable choice of the parametrization of $\mathbf{C}$ and if all the single bodies of the system are symmetric, $\dot{\mathbf{M}} - 2\mathbf{C}$ is skew symmetric.
- The matrix $\mathbf{D}$ is positive definite.

In [51.20] the mathematical model written with respect to the Earth-fixed-frame vehicle position and the manipulator end-effector can be found. However, it must be noted that, in this case, a six-dimensional manipulator is considered in order to have a square Jacobian to work with; moreover, kinematic singularities need to be avoided.

The equations of motion of UVMSs in matrix form presented in (51.34) are formally similar to the equations of motion of ground fixed manipulators (Chap. 3) for which a wide control literature exists. This has suggested a suitable translation/implementation of existing control algorithms. However, some differences, crucial from the control aspect, need to be underlined. UVMSs are complex systems characterized by several strong constraints:

- Uncertainty in the model knowledge, mainly due to the poor knowledge about the hydrodynamic effects
- The complexity of the mathematical model
- The kinematic redundancy of the system
- The difficulty in controlling the vehicle in hovering, mainly due to poor thruster performance
- The dynamic coupling between the vehicle and the manipulator
- The low bandwidth of the sensor readings.

In 1996 *McLain* et al. [51.63] presented a control law for UVMSs with some interesting experimental results conducted at the Monterey Bay Aquarium Research Institute (MBARI). A one-link manipulator was mounted on the OTTER vehicle controlled in all six DOFs by means of eight thrusters. A coordinated control system was then implemented to improve the tracking errors of the end effector.

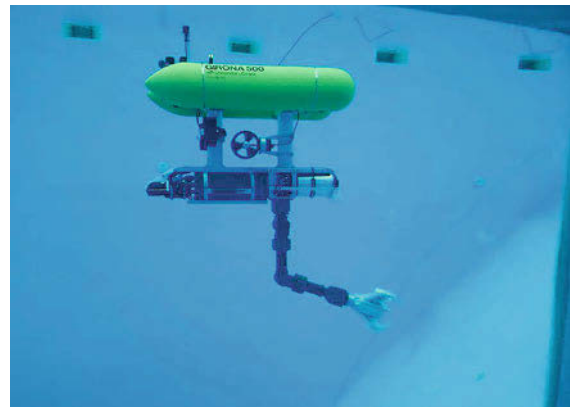


Fig. 51.8 An underwater vehicle–manipulator system (courtesy of FP7-TRIDENT Project Consortium)

The monograph [51.64] is focused on the modeling and control issues of such systems, and can be considered as a reference for further reading. Moreover, interaction with the environment is also discussed.

Currently, remotely operated manipulator are standard equipment for several underwater ROVs. However, autonomous manipulation is still a research challenge. The SAUVIM vehicle, developed at the Autonomous Systems Laboratory, University of Hawaii, one of the first semiautonomous underwater vehicle manipulator systems. Similar research projects, ALIVE and TRIDENT were funded by the Framework Program of the European Community [51.2]. Figure 51.8 reports the UVMS developed under the TRIDENT project [51.3]. VIDEO 89 shows a sampling performed with the Nereus arm.

51.2.9 Fault Detection/Tolerance

Generally, AUVs must operate over long periods of time in unstructured environments in which an undetected failure could cause the loss of the vehicle. Failure detection and a fault-tolerant strategy are required to determine whether a mission must be terminated in the safest manner possible or if the vehicle can continue in a diminished capacity. An example is the case of the arctic mission of Theseus [51.65].

In the case of the use of ROVs, a skilled human operator is in charge of commanding the vehicle; a failure detection strategy is then of help in the human decision-making process. Based on the information detected, the operator can decide on vehicle rescue or to terminate the mission by, e.g., turning off a thruster.

Fault detection is the process of monitoring a system in order to recognize the presence of a failure; fault isolation or diagnosis is the capability to determine which specific subsystem is subject to failure. Often in the literature there is a certain overlap in the use of these terms. Fault tolerance is the capability to complete the mission in the case of the failure of one or more subsystems; it is also known as fault control, fault accommodation, or control reconfiguration. In the following the terms fault detection/tolerance will be used.

The characteristics of a fault detection scheme are the capability to isolate the detected failure, sensitivity in terms of the magnitude of the failure that can be detected, and robustness in the sense of the capability to continue working properly in non-nominal conditions. The requirements of a fault-tolerant scheme are reliability, maintainability, and survival. The common concept is that, to overcome the loss of capability due to a failure, a kind of redundancy is required in the system.

In this section, a survey of existing fault-detection and fault-tolerant schemes for underwater vehicles is presented. For these specific systems, if proper strategies are applied, a hardware/software (HW/SW) sensor or thruster failure can be successfully handled. In some conditions, the fault-detection scheme must also be able to diagnose some external abnormal working conditions such as a multipath phenomena affecting the echo-sounder system. It is worth noticing that, for autonomous systems such as AUVs, space systems or aircraft, a fault-tolerant strategy is necessary to safely recover the damaged vehicle and, obviously, there is no *panic button* in the sense that the choice of turning off the power or activating some kind of brake is not available.

Most fault-detection schemes are model based [51.66, 67] and consider the dynamic relationship between the actuators and vehicle behavior or the specific input–output thruster dynamics. In general, fault-detection/tolerance theory has been applied to the specific case of the underwater environment even if only a few papers report experimental results; see [51.68] for a survey on this topic.

Most fault-tolerant schemes consider a thruster-redundant vehicle that, after a fault has occurred in one of its thrusters, is still actuated in six DOFs. Based on this assumption a reallocation of the desired forces on the vehicle over the working thrusters is performed [51.69]. Of interest is also the study of reconfiguration strategies if the vehicle becomes underactuated.

Possible Failures

Underwater vehicles are currently equipped with several sensors in order to provide information about their localization and velocity. The problem is not easy. No single, reliable sensor is available that gives the required position/velocity measurement, or information about the environment such as the presence of obstacles. For this reason the use of sensor fusion by, e.g., a Kalman filtering approach, is a common technique to provide the controller with the required variables. This structural redundancy can be used to provide fault-detection capabilities to the system.

For each of the sensors listed in Sect. 51.2.2 failure can consist of an output of zero if, e.g., there is an electrical trouble, or a loss of meaning. It can be considered as sensor failure also an external disturbance such as a multipath reading of the sonar that can be interpreted as a sensor fault and correspondingly detected.

Thruster blocking occurs when a solid body is present between the propeller blades. It can be checked by monitoring the current required by the thruster. This was observed, e.g., during the Antarctic mission of Romeo [51.70], in that case caused by a block of ice. During the same mission a thruster also flooded with water. The consequence was an electrical dispersion, causing an increasing blade rotation velocity and thus a thruster force higher then desired.

A possible consequence of different failures of the thrusters is the zeroing of the blade rotation. The thruster in question thus simply stops working. This has been intentionally experienced during experiments with, e.g., ODIN [51.67, 69], Roby 2 [51.66], and Romeo [51.70].

Other failures include a hardware/software crash or the occurrence of fin sticking or loss. A very common type of failure involves the loss of electrical isolation due to seawater intrusion into underwater electrical cables or connectors. Such a condition can be detected through a technique called ground-fault monitoring. Should this occur, electrical power must be removed from the affected device.

51.2.10 Multiple Underwater Vehicles

A growing research effort has recently been devoted to developing strategies to design coordinated control for underwater vehicles. The use of multiple AUVs, in fact, might improve overall mission performance as well as provide greater tolerance to failures. Specific applications of this method in the underwater environment might include the naval mine countermeasure problem, harbor monitoring, and inspection, exploration, and mapping of large areas. AUVs might be coordinated with one or more surface vessels or connected to ground or aerial vehicles to form a coordinated network of heterogeneous autonomous robots.

Beside several institutions that have developed simulation packages for multiple-AUV operations, the use of real multiple AUVs is being considered for the adaptive sampling and forecasting plan of the Autonomous Ocean Sampling Network, formed by several research institutions such as (for the robotic components) Caltech, MBARI, Princeton, and WHOI [51.71]. Adaptive sampling is also investigated within the project Adaptive Sampling and Prediction (ASAP) [51.72]. The Australian National University is currently working on a shoal of small, autonomous robots, named Serafina [51.73]. At Instituto Superior Técnico (IST), work is ongoing on the coordination between an AUV and a catamaran [51.74], i. e., a multirobot system constituted by heterogeneous autonomous vehicles. In [51.75] the cooperation between AUVs and a sensor network is experimentally investigated. VIDEO 323 and VIDEO 94 show two examples of multi-vehicle bathymetry mission and multi-vehicle patrolling, respectively.

51.3 Applications

Underwater robots currently play prominent roles in a number of scientific, commercial, and military tasks. Remotely teleoperated vehicles are very well established in all these areas, and are becoming increasingly automated to relieve the burden on human operators and to improve performance. Increasingly, autonomous underwater vehicles are finding application in these areas as well. Presently, AUVs are used almost exclusively for survey work, but sampling and other intervention tasks are becoming more feasible. Additionally, the line between ROVs and AUVs continues to blur, as systems that have the best properties of both evolve.

The offshore oil and gas industry relies heavily on ROVs for installation, inspection, and servicing of platforms, pipelines, and subsea production facilities. As the search for oil and gas goes deeper, this trend can only continue. The Marine Technology Society estimates that there are over 435 *work-class* ROVs operating in the commercial offshore industry today. AUVs are now beginning to appear in the commercial offshore industry for survey tasks, and concepts for hybrid systems that can perform intervention tasks are now appearing. The goal is not only for these robotic vehicles to replace human divers or human-occupied vehicles, but to enable an entire new generation of subsea equipment that is serviced without intervention by drill ships or other heavy-lifting vessels. This holds the prospect of greatly reduced cost.

Scientific demand for ROVs and AUVs is also increasing dramatically. Scientific applications for ROVs include survey, inspection, and sampling tasks previously performed by human-occupied submersibles or towed vehicles. While ROVs operating for science are not nearly as numerous as those in the offshore oil and gas industry, they are becoming commonplace. Most nations involved in global seafloor studies have several vehicles. Like the vehicles for the commercial offshore sector, these vehicles are becoming increasingly automated. High-quality electronic imaging, including high-definition television, is becoming increasingly common. Scientific ROVs are now equipped with sophisticated sampling devices for sampling animals, microbes, caustic hydrothermal vent fluids, and a variety of rock samples. Moreover, ROVs are also used to deploy and operate seafloor experiments, which can involve difficult tasks such as drilling and delicate placement of instruments.

ROVs have also emerged as powerful tools for investigating underwater shipwrecks and other cultural sites. Applications include forensic investigations of modern shipwrecks to determine the cause of sinking, archaeology, and salvage. For archaeology, the goals are the same as for excavation on land: detailed mapping followed by careful excavation. Beyond diver depths, ROVs are the preferred method for these investigations. Great progress has been made in the detailed mapping phase, and capabilities for excavation are evolving. Unfortunately, the same technology also opens the possibility for shipwrecks to be looted for financial gain, which usually results in the loss of the most valuable historical information.

After a long period of skepticism, AUVs are now accepted for scientific tasks. Presently, AUVs most often perform mapping tasks while tended by a vessel. Specific mapping tasks include seafloor bathymetry, sidescan sonar imaging, magnetic field mapping, hydrothermal vent localization, and photo surveys. AUVs have been shown to improve productivity and data quality compared to towed and tethered systems. They have also operated in environments where no other means of gathering data is possible, such as under ice shelves. Likewise, the increasing availability of sophisticated in situ chemical sensors, biological sensors, and mass spectrometers now allows AUVs to build spatial and temporal maps of environmental features that could pre-

Avidbots Ex1016
Page 1327 of 2154

viously only be studied by bringing samples back to the laboratory. AUV systems that can dock to subsea nodes to recharge batteries, offload data and receive new instructions are presently being tested.

The military has always been a leader in the development of underwater robotic capabilities. They pioneered ROVs for tasks such as recovering test weapons and deep-sea salvage, and present-day commercial and scientific ROVs have descended directly from these early systems. Likewise, military interests are presently pushing AUV technology very hard. Many different countries operate AUVs for military surveys, gathering environmental data as well as searching for hazards such as mines. An operational success was achieved in surveying for mines in the Persian Gulf harbor of Umm Qasr using REMUS vehicles. AUVs in development will not only be able to detect mines, but to disable them. Bolder, more innovative concepts are also in development. These include networks of AUVs that can act as extensions of conventional surface vessels and submarines, enabling surveillance over wide areas for extended periods of time at costs far less than could be achieved with conventional surface vessels, submarines, and aircraft. These developments will rely on improvements in acoustic communications, energy systems, sensors, and onboard intelligence that will likely find their way into commercial and scientific practice.

51.4 Conclusions and Further Reading

The underwater environment is extremely hostile for human engineering activities. In addition to high pressures and hydrodynamic forces that are both nonlinear and unpredictable, water is not an appropriate media for electromagnetic communication except at short ranges. This pushes underwater technology to rely on acoustic communication and positioning systems that are characterized by low bandwidth. On the other hand, the ocean is extremely important for numerous human activities from the commercial, cultural, and environmental points of view.

Research on underwater robotic applications is active both from the technological and methodological aspects. The power endurance of commercially AUVs is currently up to 50 h; this will increase as energy-storage devices improve. Improved energy and power capability will enable longer missions, higher speeds, or better/additional sensors such as, e.g., more powerful lighting for underwater video/photography. The current trend for the price of AUVs prices is downward, with more and smaller research institutions building or buying AUVs to enrich their research results; moreover the setup of multiple-AUV systems is becoming cost effective. The goal is to develop fully autonomous, reliable, robust, decision-making AUVs.

There are a number of technology issues that are needed in order to improve AUV capabilities: to increase the underwater bandwidth of current acoustic modems, to increase onboard power to handle larger tools and interact more strongly with the environment, to create AUVs with significant hovering capability to allow better interaction, and to enable easier launch and recovery.

In the near future, the ROV/AUV dichotomy will likely become less prominent, with a variety of systems appearing that have attributes of both systems:

- For offshore oil and gas intervention tasks, a vehicle could transit to the work site as a self-powered, fully autonomous vehicle, then dock to the work site. Utilizing energy and communications infrastructure at the work site, the vehicle could then be operated much like a conventional ROV.
- Battery-operated ROVs can communicate with the surface by very lightweight fiber-optic links, enabling the mobility of an AUV but with a high-bandwidth connection to skilled human operators for complex intervention or scientific sampling tasks.
- Acoustic and optical data links can provide moderate to high communication bandwidths over short ranges, enabling human supervision without any tether restrictions. At longer ranges, more modest acoustic bandwidths are available.

These developments make marine robotics a challenging engineering problem with strong connections to several engineering domains. Sending an autonomous vehicle into an unknown and unstructured environment with limited online communication requires some onboard intelligence and the ability for the vehicle to react in a reliable way to unexpected situations.

A major challenge concerning underwater robotics is the interaction with the environment by means of one or more manipulators. Autonomous UVMSs are still the object of research; the current trend is in developing the first semiautonomous robotic devices, which

might be acoustically operated; moreover, if physically possible, the capability to dock to the structure where the intervention is needed might significatively simplify the control. The final aim might be to develop a completely autonomous UVMS, able to localize the intervention site, recognize the task to be performed, and act on it without docking to the station and without human intervention. This might make it possible to perform missions that are currently impossible such as autonomous archaeological intervention at deep sites. This would also enable the oil and gas industry to significatively decrease costs and risks to humans.

The Marine Systems Simulator [51.76] is a Matlab/Simulink library and simulator for marine systems. It includes models for ships, underwater vehicles, and floating structures. The library also contains guidance, navigation, and control (GNC) blocks for real-time simulation. A numerical simulator including also the presence of a manipulator has been developed under the TRIDENT project [51.77].

For further reading on the topic of underwater systems, the reader is referred to several survey articles, including [51.6, 29, 35, 38, 68]. Additionally, several journals cover oceanic engineering topics, including robotics aspects. A variety of symposia and workshops have been held on a regular basis. Some books/monographs treating marine robotics are [51.4, 8, 9, 17, 64].

Video-References

VIDEO 87 Dive with REMUS
available from http://handbookofrobotics.org/view-chapter/51/videodetails/87

VIDEO 88 Underwater vehicle Nereus
available from http://handbookofrobotics.org/view-chapter/51/videodetails/88

VIDEO 89 Mariana Trench: HROV Nereus samples the Challenger Deep seafloor
available from http://handbookofrobotics.org/view-chapter/51/videodetails/89

VIDEO 90 REMUS SharkCam: The hunter and the hunted
available from http://handbookofrobotics.org/view-chapter/51/videodetails/90

VIDEO 92 The Icebot
available from http://handbookofrobotics.org/view-chapter/51/videodetails/92

VIDEO 94 Two underwater Folaga vehicles patrolling a 3-D area
available from http://handbookofrobotics.org/view-chapter/51/videodetails/94

VIDEO 267 Adaptive L1 depth control of a ROV
available from http://handbookofrobotics.org/view-chapter/51/videodetails/267

VIDEO 268 Saturation based nonlinear depth and yaw control of an underwater vehicle
available from http://handbookofrobotics.org/view-chapter/51/videodetails/268

VIDEO 323 Multi-vehicle bathymetry mission
available from http://handbookofrobotics.org/view-chapter/51/videodetails/323

VIDEO 324 Neptus command and control infrastructure
available from http://handbookofrobotics.org/view-chapter/51/videodetails/324

References

51.1 J. Yuh, S.K. Choi, C. Ikehara, G.H. Kim, G. McMurty, M. Ghasemi-Nejhad, N.N. Sarkar, K. Sugihara: Design of a semi-autonomous underwater vehicle for intervention missions (SAUVIM), IEEE Int. Symp. Underw. Technol. (1998) pp. 63–68

51.2 P. Marty: ALIVE: An autonomous light intervention vehicle, Adv. Technol. Underw. Veh. Conf., Oceanol. Int. (2004)

51.3 M. Prats, J.C. Garcia, S. Wirth, D. Ribas, P.J. Sanz, P. Ridao, N. Gracias, G. Oliver: Multipurpose autonomous underwater intervention: A systems integration perspective, 20th IEEE Mediterr. Conf. Contr. Autom., Barcelona (2012) pp. 1379–1484

51.4 T.I. Fossen: *Handbook of Marine Craft Hydrodynamics and Motion Control* (Wiley, New York 2011)

51.5 S. Bennett: A brief history of automatic control, IEEE Control Syst. Mag. **16**(3), 17–25 (1996)

51.6 J. Yuh, M. West: Underwater robotics, J. Adv. Robotics **15**(5), 609–639 (2001)

51.7 SNAME: *Nomenclature for Treating the Motion of a Submerged Body Through a Fluid*, Techn. Res. Bull. (SNAME, New York 1952) pp. 1–5

51.8 T.I. Fossen: *Guidance and Control of Ocean Vehicles* (Wiley, New York 1994)

51.9 O.M. Faltinsen: *Sea Loads on Ships and Offshore Structures* (Cambridge Univ. Press, Cambridge 1990)

51.10 J. Yuh: Modeling and control of underwater robotic vehicles, IEEE Trans. Syst. Man Cybern. **20**, 1475–1483 (1990)

51.11 T.I. Fossen, A. Ross: *Guidance and Control of Unmanned Marine Vehicles*, IEEE Control Engineering (Wiley, Chichester 1999) pp. 23–42

51.12 W.E. Cummins: *The impulse response function and ship motions, Techn. Rep. 1661* (DTIC, Washington 1962)

51.13 T.F. Ogilvie: Recent progress towards the understanding and prediction of ship motions, 5th Symp. Nav. Hydrodyn. (1964) pp. 3–79

51.14 T. Perez, T.I. Fossen: Time-domain models of marine surface vessels for simulation and control design based on seakeeping computations, 7th Conf. Manoeuvring Control Mar. Craft, (IFAC) (2006)

51.15 T.I. Fossen: A nonlinear unified state-space model for ship maneuvering and control in a seaway, J. Bifurc. Chaos **15**(9), 2717–2746 (2005)

51.16 M. Nahon: Determination of undersea vehicle hydrodynamic derivatives using the USAF, Datcom, Proc. Ocean. Conf., Victoria (1993) pp. 283–288

51.17 J.N. Newman: *Marine Hydrodynamics* (MIT, Cambridge 1977)

51.18 A.J. Healey, D. Lienard: Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles, IEEE J. Ocean. Eng. **18**, 327–339 (1993)

51.19 B. Stevens, F. Lewis: *Aircraft Control and Simulations* (Wiley, New York 1992)

51.20 I. Schjølberg, T.I. Fossen: Modelling and control of underwater vehicle-manipulator systems, 3rd Conf. Manoeuvring Control Mar. Craft (IFAC), Southampton (1994) pp. 45–57

51.21 E.A. de Barros, A. Pascoal, E. de Sea: Progress towards a method for predicting AUV derivatives, 7th Conf. Manoeuvring Control Mar. Craft (IFAC), Lisbon (2006)

51.22 Monterey Bay Aquarium Research Institute: http://www.mbari.org

51.23 N.P. Fofonoff, R.C. Millard: *Algorithms for Computation of Fundamental Properties of Seawater, UNESCO Tech. Pap. Mar. Sci. No. 44* (UNESCO, Paris 1983)

51.24 Autonomous Systems Laboratory, University of Hawaii: http://www.eng.hawaii.edu/~asl/

51.25 R. Eustice, H. Singh, J.J. Leonard, M. Walter: Visually mapping the RMS Titanic: Conservative covariance estimates for SLAM information filters, Int. J. Robotics Res. **25**(12), 1223–1242 (2006)

51.26 D.A. Smallwood, L.L. Whitcomb: Adaptive identification of dynamically positioned underwater robotic vehicles, IEEE Trans. Control Syst. Technol. **11**(4), 505–515 (2003)

51.27 S. Zhao, J. Yuh: Experimental study on advanced underwater robot control, IEEE Trans. Robotics **21**(4), 695–703 (2005)

51.28 S. Majumder, S. Scheding, H.F. Durrant-Whyte: Multisensor data fusion for underwater navigation, Robotics Auton. Syst. **35**(2), 97–108 (2001)

51.29 J.C. Kinsey, R.M. Eustice, L.L. Whitcomb: A survey of underwater vehicle navigation: Recent advances and new challenges, 7th Conf. Manoeuvring Control Mar. Craft (IFAC), Lisbon (2006)

51.30 D.R. Yoerger, J.G. Cooke, J.J. Slotine: The influence of thruster dynamics on underwater vehicle behavior and their incorporation into control system design, IEEE J. Ocean. Eng. **15**, 167–178 (1990)

51.31 A.J. Healey, S.M. Rock, S. Cody, D. Miles, J.P. Brown: Toward an improved understanding of thruster dynamics for underwater vehicles, IEEE J. Ocean. Eng. **20**(4), 354–361 (1995)

51.32 L. Bachmayer, L.L. Whitcomb, M.A. Grosenbaugh: An accurate four-quadrant nonlinear dynamical model for marine trhusters: Theory and experimental validation, IEEE J. Ocean. Eng. **25**, 146–159 (2000)

51.33 T.I. Fossen, M. Blanke: Nonlinear output feedback control of underwater vehicle propellersusing feedback form estimated axial flow velocity, IEEE J. Ocean. Eng. **25**(2), 241–255 (2000)

51.34 W.P.A. Van Lammeren, J. van Manen, M.W.C. Oosterveld: The wageningen B-screw series, Trans. SNAME **77**, 269–317 (1969)

51.35 T.I. Fossen, T.I. Johansen: A survey of control allocation methods for ships and underwater vehicles, 14th IEEE Mediterr. Conf. Control Autom., Ancona (2006) pp. 1–6

51.36 M. Stojanovic, J. Preisig: Underwater acoustic communication channels: Propagation models and statistical characterization, IEEE Commun. Mag. **47**(1), 84–89 (2009)

51.37 D. Pompili, I. Akyildiz: Overview of networking protocols for underwater wireless communications, IEEE Commun. Mag. **47**(1), 97–102 (2009)

51.38 K.P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru: Control architecture for autonomous underwater vehicles, IEEE Control Syst. **17**, 48–64 (1997)

51.39 P. Oliveira, A. Pascoal, V. Silva, C. Silvestre: Mission control of the MARIUS AUV: System design, implementation, and sea trials, Int. J. Syst. Sci. **29**(10), 1065–1080 (1998)

51.40 D. Brutzman, M. Burns, M. Campbell, D. Davis, T. Healey, M. Holden, B. Leonhardt, D. Marco, D. McClarin, B. McGhee: NPS Phoenix AUV software integration and in-water testing, Proc. IEEE Symp. Auton. Underw. Veh. Technol. (AUV) (1996) pp. 99–108

51.41 A.P. Aguiar, A.M. Pascoal: Dynamic positioning and way-point tracking of underactuated AUVs in the presence of ocean currents, Int. J. Control **80**(7), 1092–1108 (2007)

51.42 A.A. Bennett, J.J. Leonard: A behavior-based approach to adaptive feature detection and following with autonomous underwater vehicles, IEEE J. Ocean. Eng. **25**(2), 213–226 (2000)

51.43 M. Breivik, T.I. Fossen: Principles of guidance-based path following in 2D and 3D, 44th IEEE Conf. Decis. Control 8th Eur. Control Conf., Sevilla (2005)

51.44 F.A. Papoulias: Bifurcation analysis of line of sight vehicle guidance using sliding modes, Int. J. Bifurc. Chaos **1**(4), 849–865 (1991)

51.45 R. Rysdyk: UAV path following for constant line-of-sight, Proc. 2nd AIAA Unmanned Unltd. Syst. Technol. Oper. Aerosp., San Diego (2003)

Avidbots Ex1016
Page 1330 of 2154

51.46 M.D. Feezor, F.Y. Sorrel, P.R. Blankinship, J.G. Bellingham: Autonomous underwater vehicle homing/docking via electromagnetic guidance, IEEE J. Ocean. Eng. **26**(4), 515–521 (2001)

51.47 D. Wettergreen, A. Zelinsky, C. Gaskett: Autonomous guidance and control for an underwater robotic vehicle, Int. Conf. Field Serv. Robotics (1999)

51.48 G. Antonelli, S. Chiaverini, R. Finotello, R. Schiavon: Real-time path planning and obstacle avoidance for RAIS: An autonomous underwater vehicle, IEEE J. Ocean. Eng. **26**(2), 216–227 (2001)

51.49 J.C. Hyland, F.J. Taylor: Mine avoidance techniques for underwater vehicles, IEEE J. Ocean. Eng. **18**, 340–350 (1993)

51.50 D.R. Yoerger, J.J. Slotine: Robust trajectory control of underwater vehicles, IEEE J. Ocean. Eng. **10**, 462–470 (1985)

51.51 R. Cristi, F.A. Pappulias, A. Healey: Adaptive sliding mode control of autonomous underwater vehicles in the dive plane, IEEE J. Ocean. Eng. **15**(3), 152–160 (1990)

51.52 D.B. Marco, A.J. Healey: Command, control and navigation experimental results with the NPS ARIES AUV, IEEE J. Ocean. Eng. **26**(4), 466–476 (2001)

51.53 M. Caccia, G. Veruggio: Guidance and control of a reconfigurable unmanned underwater vehicle, Control Eng. Prac. **8**(1), 21–37 (2000)

51.54 G. Antonelli, F. Caccavale, S. Chiaverini, G. Fusco: A novel adaptive control law for underwater vehicles, IEEE Trans. Control Syst. Technol. **11**(2), 221–232 (2003)

51.55 G. Antonelli: On the use of adaptive/integral actions for 6-degrees-of-freedom control of autonomous underwater vehicles, IEEE J. Ocean. Eng. **32**(2), 300–312 (2007)

51.56 M. Erol-Kantarci, H.T. Mouftah, S. Oktug: A survey of architectures and localization techniques for underwater acoustic sensor networks, IEEE Commun. Surv. Tutor. **13**(3), 487–502 (2011)

51.57 A. Bahr, J.J. Leonard, M.F. Fallon: Cooperative localization for autonomous underwater vehicles, Int. J. Robotics Res. **28**(6), 714–728 (2009)

51.58 S.B. Williams, I. Mahon: A terrain-aided tracking algorithm for marine systems. In: *Field and Service Robotics*, (Springer, Berlin, Heidelberg 2006) pp. 93–102

51.59 M. Dunbabin, P. Corke, G. Buskey: Low-cost vision-based AUV guidance system for reef navigation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2004) pp. 7–12

51.60 J.J. Leonard, H.J.S. Feder: Decoupled stochastic mapping, IEEE J. Ocean. Eng. **26**(4), 561–571 (2001)

51.61 S. Williams, G. Dissanayake, H. Durrant-Whyte: Towards terrain-aided navigation for underwater robotics, Adv. Robotics **15**(5), 533–549 (2001)

51.62 P. Newman, J. Leonard: Pure range-only sub-sea SLAM, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2003) pp. 1921–1926

51.63 T.W. McLain, S.M. Rock, M.J. Lee: Experiments in the coordinated control of an underwater arm/vehicle system, Auton. Robots **3**(2), 213–232 (1996)

51.64 G. Antonelli (Ed.): *Underwater Robots. Motion and Force Control of Vehicle-Manipulator Systems*, Springer Tracts in Advanced Robotics (Springer, Berlin, Heidelberg 2014), 3rd edn.

51.65 J.S. Ferguson, A. Pope, B. Butler, R. Verrall: Theseus AUV – Two record breaking missions, Sea Technol. Mag. **40**, 65–70 (1999)

51.66 A. Alessandri, M. Caccia, G. Veruggio: Fault detection of actuator faults in unmanned underwater vehicles, Control Eng. Prac. **7**, 357–368 (1999)

51.67 K.C. Yang, J. Yuh, S.K. Choi: Fault-tolerant system design of an autonomous underwater vehicle – ODIN: An experimental study, Int. J. Syst. Sci. **30**(9), 1011–1019 (1999)

51.68 G. Antonelli: A survey of fault detection/tolerance strategies for AUVs and ROVs. Recent advances. In: *Fault Diagnosis and Tolerance for Mechatronic Systems*, Springer Tracts in Advanced Robotics, ed. by F. Caccavale, L. Villani (Springer, Berlin, Heidelberg 2002) pp. 109–127

51.69 T.K. Podder, G. Antonelli, N. Sarkar: An experimental investigation into the fault-tolerant control of an autonomous underwater vehicle, J. Adv. Robotics **15**, 501–520 (2001)

51.70 M. Caccia, R. Bono, G. Bruzzone, G. Bruzzone, E. Spirandelli, G. Veruggio: Experiences on actuator fault detection, diagnosis and accomodation for ROVs, Int. Symp. Unmanned Untethered Submers. Technol. (2001)

51.71 E. Fiorelli, P. Bhatta, N.E. Leonard, I. Shulman: Adaptive sampling using feedback control of an autonomous underwater glider fleet, Int. Symp. Unmanned Untethered Submers. Technol. (2003)

51.72 N.E. Leonard, D.A. Paley, R.E. Davis, D.M. Fratantoni, F. Lekien, F. Zhang: Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay, J. Field Robotics **27**(6), 718–740 (2010)

51.73 S. Kalantar, U. Zimmer: Distributed shape control of homogeneous swarms of autonomous underwater vehicles, Auton. Robots **22**(1), 37–53 (2006)

51.74 A. Pascoal, C. Silvestre, P. Oliveira: Vehicle and mission control of single and multiple autonomous marine robots. In: *Advances in Unmanned Marine Vehicles*, IEEE Control Engineering, ed. by G. Roberts, R. Sutton (Peregrinus, New York 2006) pp. 353–386

51.75 M. Dunbabin, I. Vasilescu, P. Corke, D. Rus: Experiments with cooperative networked control of underwater robots. In: *Experimental Robotics*, (Springer, Berlin, Heidelberg 2008) pp. 463–470

51.76 Marine Systems Simulator: http://www.marinecontrol.org

51.77 TRIDENT project: http://www.irs.uji.es/uwsim

Avidbots Ex1016
Page 1331 of 2154

Multimedia Contents

52. Modeling and Control of Aerial Robots

Robert Mahony, Randal W. Beard, Vijay Kumar

Aerial robotic vehicles are becoming a core field in mobile robotics. This chapter considers some of the fundamental modelling and control architectures in the most common aerial robotic platforms; small-scale rotor vehicles such as the quadrotor, hexacopter, or helicopter, and fixed wing vehicles. In order to control such vehicles one must begin with a good but sufficiently simple dynamic model. Based on such models, physically motivated control architectures can be developed. Such algorithms require realisable target trajectories along with real-time estimates of the system state obtained from on-board sensor suite. This chapter provides a first introduction across all these subjects for the quadrotor and fixed wing aerial robotic vehicles.

52.1 **Overview** 1307
52.2 **Modeling Aerial Robotic Vehicles** 1309
52.2.1 Rigid Body Motion of the Airframe 1309
52.2.2 Modeling for Quadrotors 1310
52.2.3 Modeling of Fixed-Wing Airplanes 1312
52.3 **Control** 1316
52.3.1 Quadrotor Control 1316
52.3.2 Control of Fixed-Wing Aircraft 1319
52.4 **Trajectory Planning** 1324
52.4.1 Trajectory Planning for Quadrotors 1324
52.4.2 Trajectory Planning for Fixed-Wing Aircraft 1325
52.5 **Estimating the Vehicle State** 1328
52.5.1 Estimating Attitude 1328
52.5.2 Estimating Velocity and Position 1329
52.6 **Conclusion** 1330
Video-References 1331
References 1331

52.1 Overview

The term *aerial robotics* is often attributed to *Robert Michelson* [52.1], as a term to capture a new class of autonomous and intelligent small flying machines. The quest for autonomy of aerial vehicles goes back almost to the origins of powered flight. The first autopilot for a fixed wing vehicle (the Sperry autopilot) was demonstrated in 1912, only a decade after the first powered flight by the Wright brothers. As early as World War I, the potential for an autonomously stabilized and remotely controlled airplane to be an effective weapon was understood and the Curtis–Sperry flying bomb first demonstrated autonomous unmanned flight in 1918 [52.2]. Autopilot technology was perfected during the 1930s as improving aircraft technology lead to longer flight times and the need to relieve the pilot from constant attention to flight stability of the vehicle. Autonomous and radio controlled target aircraft were developed in the late 1930s and were extensively used during World War II for training. The German V-1 cruise missile, commonly known as the buzz bomb or doodlebug, was a highly successful autonomous vehicle; its relatively low impact in World War II being due to more British intelligence efforts than to the vehicles capabilities.

Following the World War II, the relative maturity of autopilot technology and airplane construction meant that the construction of unmanned fixed-wing vehicles was straightforward. Remote-controlled heli-

Avidbots Ex1016
Page 1332 of 2154

copters, such as the Gyrodyne QH-50 DASH, were developed as early as the 1960s. The Yamaha R-50 and subsequent Yamaha R-Max (Fig. 52.1), was developed during the 1980s and provides a remote controlled commercial aerial platform, with significant onboard autonomy, used primarily for agricultural applications.

Extensive development of fixed-wing autonomous vehicles was held back in the mid-20th century by the difficulty of localizing a vehicle when far from the base station. Indeed, the requirements of navigation systems for cruise missiles developed during the cold war was one of the main drivers for the development of satellite global positioning system (GPS) technology. Full operational capability of the United States GPS system with 24 satellites was declared in April 1995 and it was mandated as a dual-use system (both commercial and military) in 1996. Smaller scale commercial unmanned aerial vehicles were also constrained by the lack of robust and small-scale avionics systems. The availability of small low power computers, the development of microelectromechanical systems (MEMS) that provide affordable and robust Inertial Measurement Units (IMUs), and access to reliable GPS, lead to the beginning of the modern era of nonmilitary UAV and aerial robotic systems in the mid-1990s.

The largest class of commercial and military robotic aerial vehicles fly predominantly in obstacle free airspace. For such vehicles, once the aircraft has taken off, there is no need to employ obstacle avoidance techniques or interact with cluttered three-dimensional environments.

Definition 52.1

An *unmanned aerial vehicle* (UAV) is a system capable of sustained flight with no direct human control and able to perform a specific task.



Fig. 52.1 Autonomous aerial robotic vehicle based on the Yamaha RMAX platform developed in the University of New South Wales, Australia (after [52.3, 4])

Navigation and control of an unmanned aerial vehicle is typically based on stabilizing a reference heading and altitude derived from the error between preset GPS way-points and the present vehicle position using a classical autopilot control system. The sensor suite used for control and navigation is usually a GPS and IMU along with barometric pressure. These sensor suites can be thought of as *proprioceptive* sensors, since they measure the internal state of the vehicle without reference to the external world. The kinds of tasks that UAVs are ideal for include high level surveillance and sensing tasks where the payload sensing suite is separate from the vehicle systems, with applications in agriculture, environmental monitoring, geophysical surveys, search and rescue, and security surveillance, as well as deployment of materials, where once again the payload is separate from the vehicle system, with applications in agriculture, search and rescue, and of course military missions. Such systems are a natural development of traditional flight technology and much of the research and development that goes into these vehicles is undertaken in aerospace departments in universities and aerospace companies.

Definition 52.2

An *aerial robotic vehicle* is an aerial vehicle capable of autonomously interacting with a complex dynamic three-dimensional environment and achieving complex environment-dependent goals.

The nature of the interaction with the environment for an aerial robotic vehicle, means that the presence of *exteroceptive* sensors, those that sense the environment around the vehicle, and how these sensor inputs are integrated into the vehicle guidance is the defining property of such a vehicle. Typical examples of exteroceptive sensors for aerial robotic vehicles are vision systems, laser range finders, acoustic sensors, etc., and external sensor systems such as VICON [52.5] and Optitrack [52.6] systems. The close coupling between the goal of the vehicle, its sensing suite, and the dynamic nature of the environment makes the simple way-point navigation control architecture of an unmanned aerial vehicle unsuitable for aerial robotic vehicles. The kinds of tasks that aerial robotic vehicles are ideal for are small-scale interactive tasks such as inspection of civil infrastructure including dam walls, girders of bridges, industrial pressure vessels; and surveillance tasks such as, inspection of damaged or burning buildings, monitoring of crowds, etc. Future applications may involve aerial manipulation including repair of infrastructure, and material handling in the construction and agricultural industries. Although the flight technology remains important, the sensing and control tasks become cru-

Avidbots Ex1016
Page 1333 of 2154

cial in such applications and the development that goes into these vehicles tends to be undertaken in robotics departments in universities and in the growing field of new aerial-robotics start-up companies. The ability to move in three-dimensional space brings new research challenges to the robotics community compared to the wheeled mobile robot technology that has motivated mobile robotics research over the last couple of decades.

The term *unmanned aerial system* (UAS) is useful to refer to the combination of the infrastructure, human interfaces, aerial platform, and sensing and control subsystems. It is clear that certain vehicles may spend time operating in either UAV mode or aerial robotic mode and the terms are more useful to describe the way of thinking about the operation of a system rather than classifying a vehicle.

52.2 Modeling Aerial Robotic Vehicles

There is already a vast range of aerial vehicle and UAV designs in existence, and there are even more ideas under development by companies and research groups as we write. In one short chapter, we must limit the scope of our presentation to a couple of the key examples. There are two major categories of vehicles that are used as platforms for aerial robotics: small-scale rotor vehicles such as the quadrotor, hexacopter, or helicopter, and fixed wing vehicles. The word *quad* is derived from the Latin *quadrangulum* for a four-sided figure. *Rotor* is derived from the Latin *rotationem* for a rotating object. However, *hexa* is derived directly from Greek for six. *Heli* is derived from ancient Greek *helikos* for spiral, while the term *copter* is derived from the modern term *helicopter*, a combination of *helikos* with the ancient Greek *pteron* for wing. Based on this etymology, we propose the terms *quadrotor* and *hexacopter* to describe the most common modern rotary wing aerial robotic vehicles in preference to the etymological questionable terms quadcopter and hexarotor that are also in common usage in the literature. We will present modeling and control material relevant to these two categories, focusing on the basic structure of the model. We will also focus on vehicles in 500 g–4 kg weight range as this corresponds to the primary general *robotics* applications where avionics and sensor suites take up the majority of the payload. Other classes of autonomous aerial vehicles that we are unable to cover in this chapter include lighter-than-air vehicles such as blimps and balloons, flapping wing vehicles, ducted fans, and rockets.

52.2.1 Rigid Body Motion of the Airframe

Heavier-than-air aerial robotic vehicles consist of a rigid airframe coupled with aerodynamic mechanisms to generate lift and thrust. For vehicles in the weight range considered, the compactness and structural integrity of the airframe means that there is little flexing of the airframe in normal operation and the rigid-body assumption is an effective model. In this case, the vehicle model can be developed based on rigid-body dynamics with exogenous forces and torque generated by an aerodynamic model.

Let $\{e_1, e_2, e_3\}$ be the coordinate axis unit vectors $e_1 = (1,0,0)^{\mathrm{T}}$, $e_2 = (0,1,0)^{\mathrm{T}}$, and $e_3 = (0,0,1)^{\mathrm{T}}$ without specification of a frame of reference. Let $\{B\}$ be a (right-hand) body fixed frame for the airframe with unit vectors $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$ where these vectors are the axes of frame $\{B\}$ with respect to frame $\{A\}$ as shown in Fig. 52.2. Note that the convention for choice of axes shown in Fig. 52.2 is the convention common in the aerial robotics community [52.7–14], but is opposite to the usual convention in the aerospace community where the $\boldsymbol{b}_2$ and $\boldsymbol{b}_3$ axes are usually reversed in order so that $\boldsymbol{b}_3$ points down in the direction of gravity. In this chapter, we have chosen to follow the robotic community convention and this will lead to some nonconventional definitions for the modeling of fixed-wing airplanes. We will discuss this further in the specific sections on fixed-wing airplane modeling and control.

The orientation of the rigid body is given by a rotation matrix ${}^{A}R_{B} = R = [\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3] \in \mathrm{SO}(3)$ in the spe-

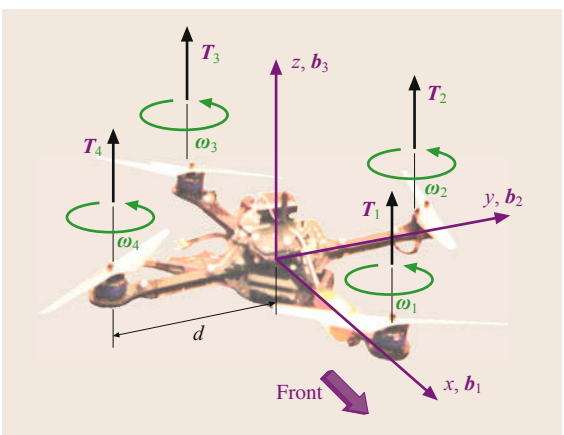


Fig. 52.2 The body-fixed frame and the directions of rotation for the propellers

cial orthogonal group. We have $\boldsymbol{b}_1 = Re_1$, $\boldsymbol{b}_2 = Re_2$, $\boldsymbol{b}_3 = Re_3$ by construction.

We will use *Z-X-Y* Euler angles to model this rotation as shown in Fig. 52.3. Note that this is not the normal roll-pitch-yaw convention used in aerospace. To get from $\{A\}$ to $\{B\}$, we first rotate about e_3 by the yaw angle, ψ, and we will call this intermediary frame $\{D\}$ with a basis $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3\}$ where $\boldsymbol{d}_i$ is expressed with respect to frame $\{A\}$. This is followed by a rotation about the x-axis in the rotated frame through the roll angle, ϕ, to intermediary frame $\{E\}$, followed by a third pitch rotation about the new y-axis through the pitch angle θ which results in the body-fixed triad $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$. The associated rotation matrix is

$$R = \begin{pmatrix} c_\psi c_\theta - s_\phi s_\psi s_\theta & -c_\phi s_\psi & c_\psi s_\theta + c_\theta s_\phi s_\psi \\ c_\theta s_\psi + c_\psi s_\phi s_\theta & c_\phi c_\psi & s_\psi s_\theta - c_\psi c_\theta s_\phi \\ -c_\phi s_\theta & s_\phi & c_\phi c_\theta \end{pmatrix} . \tag{52.1}$$

where c and s are shorthand for cosine and sine, respectively.

Let ξ denote the position of the center of mass of the airframe in frame $\{A\}$ and assume that this is also the origin of frame $\{B\}$. Let $v \in \{A\}$ denote the linear velocity of $\{B\}$ with respect to $\{A\}$ expressed in $\{A\}$. Let $\Omega \in \{B\}$ denote the angular velocity of $\{B\}$ with respect to $\{A\}$, this time expressed in $\{B\}$. Let m denote the mass of the rigid object and $\mathbf{I} \in \mathbb{R}^{3\times3}$ denote the constant inertia matrix (expressed in the body fixed frame $\{B\}$). The rigid body equations of motion of the airframe are [52.15, 16]

$$\dot{\xi} = v \,, \tag{52.2a}$$

$$m\dot{v} = -mge_3 + RF \,, \tag{52.2b}$$

$$\dot{R} = R\Omega_\times \,, \tag{52.2c}$$

$$\mathbf{I}\dot{\Omega} = -\Omega \times \mathbf{I}\Omega + \tau \,. \tag{52.2d}$$

The notation $\Omega_\times$ denotes the skew-symmetric matrix such that $\Omega_\times v = \Omega \times v$ for the vector cross-product $\times$ and any vector $v \in \mathbb{R}^3$. The vectors $F, \tau \in \{B\}$ combine the principal nonconservative forces and torques applied to the airframe by the aerodynamics of the propulsion systems and lifting surfaces of the vehicle.

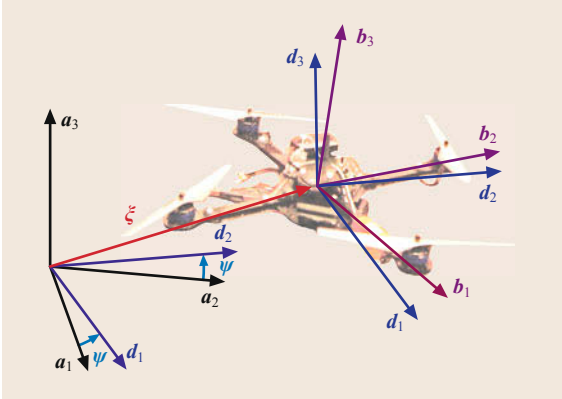


Fig. 52.3 The vehicle model. The position and orientation of the robot in the global frame are denoted by $\boldsymbol{\xi}$ and R, respectively

52.2.2 Modeling for Quadrotors

Quadrotors are currently the most popular research aerial robotics research platform because they are highly maneuverable and enable safe and low-cost experimentation in mapping, navigation, and control strategies in three dimensions. They are also arguably the simplest of the aerial robotics platforms to model for robotics applications. Early work in aerial robotics modeling and control dates back to the late 1990s and the field has remained active since [52.15–21].

In its most common form, the quadrotor vehicle is a very simple machine. It consists of four individual rotors attached to a rigid cross airframe as shown in Fig. 52.4. Control of a quadrotor is achieved by differential control of the thrust generated by each rotor. Pitch, roll, and heave (total thrust) control is straightforward to conceptualize. As shown in Fig. 52.2, rotor i rotates anticlockwise (positive about the z-axis) if i is even and clockwise if i is odd. Yaw control is obtained by adjusting the average speed of the clockwise and anticlockwise rotating rotors. The system is under-actuated and the remaining degrees of freedom corresponding to the translational velocity in the $\boldsymbol{b}_1$–$\boldsymbol{b}_2$ plane, must be controlled through the system dynamics.

The aerodynamics of rotors was extensively studied during the mid, 1900s with the development of manned helicopters and detailed models of rotor aerodynamics

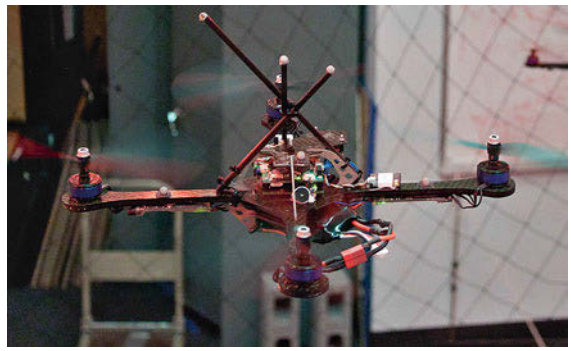


Fig. 52.4 A quadrotor made by Ascending Technologies at the University of Pennslvania with Vicon markers for state estimation

are available in the literature [52.22, 23]. Much of the detail in these aerodynamic models is useful for the design of rotor systems where the whole range of parameters (rotor geometry, profile, hinge mechanism, and much more) are fundamental to the design problem. For a typical robotic quadrotor vehicle, the rotor design is a question of choosing one among the several available rotors from the hobby shop and most of the complexity of aerodynamic modeling is best ignored. Nevertheless, a basic level of aerodynamic theory is important to understand the particularities of the control design.

The steady-state thrust generated by a hovering rotor (i. e., a rotor that is not translating horizontally or vertically) in free air may be modeled [52.23, Sect. 2.26] as

$$T_i := C_\mathrm{T} \rho A_{r_i} r_i^2 \varpi_i^2 \,, \tag{52.3}$$

where for rotor i, A_{r_i} is the rotor disk area, r_i is the radius, ϖ_i is the angular velocity, C_T is the thrust coefficient that depends on rotor geometry and profile, and ρ is the density of air. In practice, a simple lumped parameter model

$$T_i = c_\mathrm{T} \varpi_i^2 \,, \tag{52.4}$$

is used where $c_\mathrm{T} > 0$ is modeled as a constant that can be easily determined from static thrust tests.

The reaction torque (due to rotor drag) acting on the airframe generated by a hovering rotor in free air may be modeled as [52.23, Sect. 2.30]

$$Q_i := c_\mathrm{Q} \varpi_i^2 \,, \tag{52.5}$$

where the coefficient c_Q (which also depends on A_{r_i}, r_i, and ρ) can be determined by static thrust tests.

As a first approximation, assume that each rotor thrust is oriented in the $\boldsymbol{b}_3$-axis of the vehicle, although we note that this assumption does not hold exactly for a rotor translating through the air.

The total thrust at hover T applied to the airframe is the sum of the thrusts from each individual rotor (Fig. 52.2)

$$T = \sum_{i=1}^{4} |T_i| = c_\mathrm{T} \left(\sum_{i=1}^{4} \varpi_i^2 \right) . \tag{52.6}$$

The hover thrust is the primary component of the exogenous force

$$F = Te_3 + \Delta \tag{52.7}$$

in (52.2b) where Δ comprises secondary aerodynamic forces that are induced when the assumption that the rotor is in hover is violated.

The net moment arising from the aerodynamics (the combination of the individual rotor forces) applied to the quadrotor vehicle in directions $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, and $\boldsymbol{b}_3$, respectively, are

$$\begin{aligned} \tau_1 &= c_\mathrm{T} d (\varpi_2^2 - \varpi_4^2) \,, \\ \tau_2 &= -c_\mathrm{T} d (\varpi_1^2 - \varpi_3^2) \,, \\ \tau_3 &= c_\mathrm{Q} \sum_{i=1}^{4} \sigma_i \varpi_i^2 \,, \end{aligned} \tag{52.8}$$

where d is the arm length of the quadrotor. A consequence of the structure of the thrust generation and the lack of aerodynamic lifting surfaces for a quadrotor means that one can solve for the heave and torque in a single equation in terms of the motor inputs

$$\begin{pmatrix} T \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix} = \underbrace{\begin{pmatrix} c_\mathrm{T} & c_\mathrm{T} & c_\mathrm{T} & c_\mathrm{T} \\ 0 & dc_\mathrm{T} & 0 & -dc_\mathrm{T} \\ -dc_\mathrm{T} & 0 & dc_\mathrm{T} & 0 \\ -c_\mathrm{Q} & c_\mathrm{Q} & -c_\mathrm{Q} & c_\mathrm{Q} \end{pmatrix}}_{\Gamma} \begin{pmatrix} \varpi_1^2 \\ \varpi_2^2 \\ \varpi_3^2 \\ \varpi_4^2 \end{pmatrix} . \tag{52.9}$$

Inverting this relationship provides a mapping from desired control input for the rigid-body dynamics to rotor speed set-points for the motor control.

In practice, there are additional second-order aerodynamic effects present in the thrust generation for quadrotor vehicles [52.17, 19–21]. The principle second-order aerodynamic effects that are present at low speeds are *inflow variation*, *rotor flapping*, and *induced drag*. The first of these effects reduces the heave thrust when the quadrotor is ascending and increases the thrust when it is descending due to variation in rotor inflow velocity caused by the quadrotor motion. This acts as a damping term in the vertical direction of motion for the quadrotor. The remaining two effects generate forces that oppose horizontal translation of the quadrotor; blade flapping by inclining the rotor plane of flexible blade rotors and tilting the thrust vector away from the direction of motion, and induced drag by increasing the drag on the blade of a rigid rotor advancing in the direction of motion of the quadrotor [52.21]. In practice, it is difficult to separate the difference between the various effects and it is sufficient to model them all by a single damping force

$$\Delta = -Dv \,, \tag{52.10}$$

where D is a diagonally dominant positive definite matrix. For small quadrotors with relatively rigid rotor blades designed for hover performance, that is with

Avidbots Ex1016
Page 1336 of 2154

close to ideal chord and ideal twist, these effects are far more significant than is usual in classical helicopters and the effect must be modeled to achieve good control performance. These aerodynamic effects are also important in providing key low-frequency excitation to attitude and velocity estimation in the absence of GPS [52.21, 24–26].

At higher translation speeds, a quadrotor also experiences *translational lift*, *translational drag*, and *parasitic drag*. These aerodynamic effects are associated with efficiency gains in the rotor operation associated with increased inflow velocity generated by the forward velocity of the vehicle. Such effects are extremely important in large scale manned helicopters, which spend much of their time in forward flight. For many robotics applications, however, the vehicle is in quasi-stationary or near hover flight almost all the time and these secondary aerodynamic effects may be ignored. We will not discuss them further in this chapter but refer the interested reader to recent work [52.18, 20] and any of the classical helicopter texts [52.22, 23, 27].

52.2.3 Modeling of Fixed-Wing Airplanes

The conventional model for fixed-wing airplane uses a north-east-down frame of reference. The frame of reference used in this chapter, corresponding to the common aerial robotics convention inherited from the mobile-robotics community, is a north-west-up convention. We will still need to define the usual auxiliary angles associated with fixed-wing modeling and control, in particular; the angle-of-attack, side-slip, aerodynamic-bank, flight-path, and course angles. We will use the normal conventions to define these angles. The north-west-up convention now means that flight-path angle is positive when descending and negative when ascending, and positive roll leads course angle to decrease. The careful reader can easily transform between the conventions used here and the conventional airplane modeling approach and the authors apologize for any confusion that is caused by using a single convention for the whole chapter.

The key difference in modeling fixed-wing airplanes to quadrotors is that they rely on lift generated by airflow over the wings to support them in flight. The angle of incidence of the wind on the wing is a key variable in the dynamics of the vehicle and must be modeled in the dynamics. Since the main lift force is generated by an aerodynamic process, rather than directly by a controlled input as was the case for quadrotors and blimps, the modeling and control of an airplane is somewhat more complicated.

The aerodynamics of a fixed wing vehicle are defined relative to the local wind frame $\{W\}$. Frame $\{W\}$ is chosen co-linear with the inertial frame $\{A\}$, but with linear velocity equal to the average extrinsic wind velocity with respect to the inertial frame. The wind frame is a Galilean frame (moving with constant velocity) with the property that it has zero extrinsic wind on average. Let $v_a \in \{W\}$ denote the aerodynamic velocity of the vehicle, that is the linear velocity of the vehicle in the wind frame $\{W\}$. Let $v_w \in \{A\}$ denote the average wind velocity as measured in the inertial frame $\{A\}$. Since the orientation of frames $\{A\}$ and $\{W\}$ are equal, the inertial velocity of the vehicle can be written as

$$v = v_a + v_w \in \{A\} .$$

To model the incident wind as seen by the vehicle as it flies through the air we introduce the flow frame (also termed the air frame) denoted by $\{F\} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3)$ where the orientation vectors $\{\boldsymbol{f}_i\}$ are expressed in the wind frame $\{F\}$ which is co-linear with the inertial frame $\{A\}$. The first axis of the flow frame $\boldsymbol{f}_1$ is oriented in the direction of the incident wind on the vehicle, $\boldsymbol{f}_3$ is chosen to lie in the plane of symmetry of the vehicle orthogonal to $\boldsymbol{b}_2$, and $\boldsymbol{f}_2$ makes up the right-hand frame. By construction, the velocity of the vehicle in the flow frame always lies in the direction $\boldsymbol{f}_1$.

We define two orientation matrices associated with the wind frame. The angle-of-attack (AOA) matrix ${}^B R_F = R_{\alpha,\beta}$ describes the orientation of the airflow frame $\{F\}$ with respect to the body-fixed-frame $\{B\}$ in terms of the *angle-of-attack* α, and the *side-slip angle* β [52.28] (Fig. 52.5). Using the Z–Y–X Euler angle convention (yaw by β, pitch by α, and there is no roll) we have

$$R_{\alpha,\beta} = \begin{pmatrix} c_\alpha c_\beta & -c_\alpha s_\beta & -s_\alpha \\ s_\beta & c_\beta & 0 \\ s_\alpha c_\beta & s_\alpha s_\beta & c_\alpha \end{pmatrix} . \qquad (52.11)$$

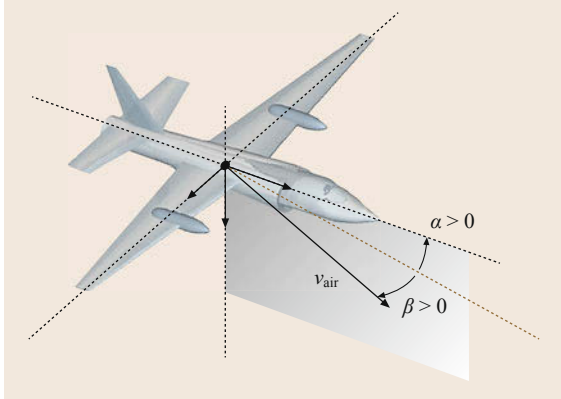


Fig. 52.5 Representation of angle-of-attack, side-slip angle, and airspeed. Here v_{air} is denoted by v_a in the text to save space

Avidbots Ex1016
Page 1337 of 2154

The second orientation matrix is the flight-path matrix ${}^{W}R_F = R_{\mu,\gamma,\chi}$ that denotes the orientation of the flow frame $\{F\}$ with respect to the wind frame $\{W\}$ in terms of the *aerodynamic bank angle* μ, the *flight path angle* γ, and the *course angle* χ. Using the *Z-Y-X* Euler angle convention (yaw by χ, pitch by γ, and roll by μ) we have

$$R_{\mu,\gamma,\chi} = \begin{pmatrix} \mathrm{c}_\gamma \mathrm{c}_\chi & \mathrm{s}_\gamma \mathrm{s}_\mu \mathrm{c}_\chi - \mathrm{c}_\mu \mathrm{s}_\chi & \mathrm{s}_\gamma \mathrm{c}_\mu \mathrm{c}_\chi + \mathrm{s}_\mu \mathrm{s}_\chi \\ \mathrm{c}_\gamma \mathrm{s}_\chi & \mathrm{s}_\gamma \mathrm{s}_\mu \mathrm{s}_\chi + \mathrm{c}_\mu \mathrm{c}_\chi & \mathrm{s}_\gamma \mathrm{c}_\mu \mathrm{s}_\chi - \mathrm{s}_\mu \mathrm{c}_\chi \\ -\mathrm{s}_\gamma & \mathrm{c}_\gamma \mathrm{s}_\mu & \mathrm{c}_\gamma \mathrm{c}_\mu \end{pmatrix}. \tag{52.12}$$

Note that the frames $\{W\}$ and $\{A\}$ are co-linear so ${}^{A}R_F = {}^{W}R_F$. It follows that ${}^{W}R_F = {}^{A}R_F = {}^{A}R_B\, {}^{B}R_F$.

Aerodynamic forces generated by lifting surfaces are proportional to the dynamic pressure [52.28]

$$\bar{Q} = \frac{1}{2}\rho |v_\mathrm{a}|^2 , \tag{52.13}$$

where ρ is the air density and $|v_\mathrm{a}|$ represents the airspeed or norm of the velocity v_a. Consider the situation where the aircraft is in normal flight, that is cruising at subsonic velocity while avoiding stall phenomenon. Let S denote the surface area of the wing (or lifting surface of the vehicle). The lift $L = F_Z^\mathrm{a}$ generated by the lifting surface is defined as the aerodynamic force oriented in the $\boldsymbol{f}_3$ direction in flow frame $\{F\}$ and can be modeled as

$$\begin{aligned} L &= \bar{Q}SC_\mathrm{L}^\alpha(\alpha + \alpha_0) + \bar{Q}SC_\mathrm{L}^\beta \beta \\ &\approx \bar{Q}S(C_\mathrm{L} + C_\mathrm{L}^\alpha \alpha) \end{aligned} \tag{52.14}$$

for sufficiently small angle of attack α (typically less that 15°). The terms $C_\mathrm{L}^\alpha(\alpha + \alpha_0)$ and $C_\mathrm{L}^\beta \beta$ are linear approximations to the lift-coefficient curves for small α and β [52.28]. The offset angle of attack α_0 is thought of as the effective angle of attack of the wing in normal level flight conditions. In particular, at normal cruising speed in level flight, $\bar{Q}SC_\mathrm{L}^\alpha \alpha_0 = mg$ is the lift required to sustain the vehicle in level flight, so that $\alpha, \beta = 0$ in level flight. The term $\bar{Q}SC_\mathrm{L}^\beta \beta$ is small and can be discarded in many applications.

The drag $D = -F_X^\mathrm{a}$ is a combination of parasitic and induced drag and is oriented in the $-\boldsymbol{f}_1$ direction.

$$\begin{aligned} D &= \bar{Q}S\left[C_\mathrm{D}^p + \kappa (C_\mathrm{L}^\alpha)^2(\alpha + \alpha_0)^2\right] \\ &\approx \bar{Q}S(C_\mathrm{D} + C_\mathrm{D}^\alpha \alpha) , \end{aligned} \tag{52.15}$$

where κ is the *Oswald coefficient* [52.28] and the second equation is obtained by ignoring α^2 terms. The constant C_D combines the parasitic drag coefficient C_D^p along with the induced drag $\kappa(C_\mathrm{L}^\alpha)^2\alpha_0^2$ due to α_0, while $C_\mathrm{D} = 2\kappa\alpha_0(C_\mathrm{L}^\alpha)^2$ is the linear coefficient for the dependence of induced drag on the angle-of-attack. It is also possible to model a contribution $\bar{Q}SC_\mathrm{D}^\beta \beta$ to the drag force, however, this is sufficiently small that it is normally ignored. There is a final lateral aerodynamic force component F_Y^a that is due to sideslip

$$F_Y^\mathrm{a} = \bar{Q}SC_Y^\beta \beta . \tag{52.16}$$

The propeller or thrust mechanism of a fixed wing vehicle generates thrust T in direction $\boldsymbol{b}_x$ in the body-fixed-frame. The total linear force $F \in \{B\}$ applied to the vehicle due to aerodynamic effects and thrust effects is

$$\begin{aligned} F &= R_{\alpha,\beta}(F_X^\mathrm{a}, F_Y^\mathrm{a}, F_Z^\mathrm{a})^\mathrm{T} + (T, 0, 0)^\mathrm{T} \\ &= \begin{pmatrix} T \\ 0 \\ 0 \end{pmatrix} + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}\mathrm{c}_\alpha \mathrm{c}_\beta - C_\mathrm{L}\mathrm{s}_\alpha \\ -C_\mathrm{D}\mathrm{s}_\beta \\ -C_\mathrm{D}\mathrm{s}_\alpha \mathrm{c}_\beta + C_\mathrm{L}\mathrm{c}_\alpha \end{pmatrix} \\ &\quad + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}^\alpha \mathrm{c}_\alpha \mathrm{c}_\beta - C_\mathrm{L}^\alpha \mathrm{s}_\alpha \\ -C_\mathrm{D}^\alpha \mathrm{s}_\beta \\ -C_\mathrm{D}^\alpha \mathrm{s}_\alpha \mathrm{c}_\beta + C_\mathrm{L}^\alpha \mathrm{c}_\alpha \end{pmatrix} \alpha \\ &\quad + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}^\beta \mathrm{c}_\alpha \mathrm{c}_\beta - C_Y^\beta \mathrm{c}_\alpha \mathrm{s}_\beta - C_\mathrm{L}^\beta \mathrm{s}_\alpha \\ -C_\mathrm{D}^\beta \mathrm{s}_\beta + C_Y^\beta \mathrm{c}_\beta \\ -C_\mathrm{D}^\beta \mathrm{s}_\alpha \mathrm{c}_\beta + C_Y^\beta \mathrm{s}_\alpha \mathrm{s}_\beta + C_\mathrm{L}^\beta \mathrm{c}_\alpha \end{pmatrix} \beta \end{aligned} \tag{52.17}$$

expressed in the flow frame.

The aerodynamic moments $\tau \in \{B\}$ in the body-fixed frame are modeled by

$$\begin{pmatrix} \tau^x \\ \tau^y \\ \tau^z \end{pmatrix} = \bar{Q}Sl \begin{pmatrix} C_x^\beta \beta + C_x^p \dfrac{c\Omega_x}{2|v_\mathrm{a}|} + C_x^r \dfrac{c\Omega_z}{2|v_\mathrm{a}|} + C_x^\delta \delta_x \\ C_y + C_y^\alpha \alpha + C_y^q \dfrac{b\Omega_y}{2|v_\mathrm{a}|} + C_y^\delta(\delta_y + \delta_y^0) \\ C_z^\beta \beta + C_z^r \dfrac{c\Omega_z}{2|v_\mathrm{a}|} + C_z^\delta \delta_z \end{pmatrix}, \tag{52.18}$$

where $(\Omega_x, \Omega_y, \Omega_z)$ are the components of the angular velocity $\Omega \in \{B\}$ and represent the roll, pitch and yaw rates of body-fixed frame, respectively; b is the wing span, c is the mean cord of the wing,

Avidbots Ex1016
Page 1338 of 2154

and $(\delta_x, \delta_y, \delta_z)$ are deflections of the airplane control surfaces (ailerons, elevator, and rudder). The constants $\{C_x^p, C_x^r, C_x^\delta, C_y, C_y^\alpha, C_y^q, C_y^\delta, C_z^\beta, C_z^r, C_z^\delta\}$ are dimensionless aerodynamic coefficients [52.28]. In normal trim conditions, the trimmed elevator deflection δ_y^0 is chosen to cancel the static moment C_y generated by the wing in level flight, $C_y^\delta \delta_y^0 = -C_y$.

Combining (52.17) and (52.18) with (52.2) yields a dynamic model for a fixed wing vehicle with minimal approximations. The resulting equations of motion are complex and can be difficult to work with directly. The most common approach to control fixed-wing vehicles based on the full model is to take this model and linearize the state equations, either along a trajectory, or in regions of state space. The resulting model is either a linear time-varying or linear parameter varying system and can be controlled using classical linear system control techniques [52.29]. Although this approach is well established in the aerospace industry it tends to hide the underlying structure of the dynamics and requires good models of the various aerodynamic parameters, a challenge for many aerial robots with airframes that are always subject to being fiddled with and having sensor packages attached externally. For aerial robotic applications it is of interest to consider a simplified but still structurally consistent model of fixed wing dynamics and use robust and simple control strategies.

Bank-to-Turn Flight Regime

In this section, we propose a model that is suitable for a wide range of fixed-wing applications where the vehicle is in normal flight mode. The approach taken uses a flight regime where the lateral acceleration required to turn the vehicle is obtained by tilting the lift generated by the wings, a strategy termed *bank-to-turn* or *coordinated turn* in the aerospace literature [52.30, 31]. A bank-to-turn maneuver is characterized by zero sideslip, $\beta = 0$, and leads to considerable simplification of the equations of motion. This is a very common mode of flying for any fixed-wing vehicle and, unless the task requires aerobatic maneuvers or the vehicle lacks control surface actuation, is the natural mode to control a UAV.

Given that accurate measurements of the roll rates Ω are available through an onboard IMU system and most UAV systems have large control surfaces relative to their size, it is possible to use high gain to dominate the attitude dynamics (52.2d) and (52.18). That is, the angular velocity $\Omega \approx \Omega^\star$ can be thought of as an input to the reduced order model (52.2a), (52.2b), (52.2c). The above discussion motivates a dynamic reduction of the system equations with new inputs $\Omega^\star$, as well as T the propeller thrust.

Based on the assumption that the vehicle is flying using bank-to-turn control and for small angle-of-attack then the approximation $\beta \approx 0$ holds and simplified dynamic equations can be derived. Using this approximation and canceling all second-order terms in α and β allows one to rewrite (52.17) as

$$F = \begin{pmatrix} T \\ 0 \\ 0 \end{pmatrix} + \bar{Q}S \begin{pmatrix} -C_\mathrm{D} \\ 0 \\ C_\mathrm{L} \end{pmatrix} + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}^\alpha - C_\mathrm{L} \\ 0 \\ C_\mathrm{L}^\alpha - C_\mathrm{D} \end{pmatrix} \alpha \,. \tag{52.19}$$

From the construction of $\{F\}$ we have ${}^v F_a = |v_\mathrm{a}| \boldsymbol{f}_1$ by definition. It follows that (52.2a) can be written as

$$\dot{\xi} = v_\mathrm{a} = |v_\mathrm{a}| R_{\mu,\gamma,\chi} e_1 \,.$$

Note that $R_{\mu,\gamma,\chi} e_1 = (\mathrm{c}_\chi \mathrm{c}_\gamma, \mathrm{s}_\chi \mathrm{c}_\gamma, -\mathrm{s}_\gamma)^\mathrm{T}$ does not depend on the aerodynamic bank angle μ. Thus, $(|v_\mathrm{a}|, \gamma, \chi)$ can be used as generalized coordinates for the velocity of the vehicle

$$\dot{\xi} = |v_\mathrm{a}| (\mathrm{c}_\chi \mathrm{c}_\gamma, \mathrm{s}_\chi \mathrm{c}_\gamma, -\mathrm{s}_\gamma)^\mathrm{T} \,. \tag{52.20}$$

Differentiating $v_\mathrm{a} = \dot{\xi}$ and rearranging yields

$$\dot{v}_\mathrm{a} = R_{\mu,\gamma,\chi} \begin{pmatrix} 1 & 0 & 0 \\ 0 & |v_\mathrm{a}| \mathrm{c}_\gamma \mathrm{s}_\mu & -|v_\mathrm{a}| \mathrm{s}_\mu \\ 0 & -|v_\mathrm{a}| \mathrm{c}_\gamma \mathrm{s}_\mu & -|v_\mathrm{a}| \mathrm{c}_\mu \end{pmatrix} \begin{pmatrix} |\dot{v}_\mathrm{a}| \\ \dot{\chi} \\ \dot{\gamma} \end{pmatrix} . \tag{52.21}$$

Consider the $(3,2)$ element of (52.12) and recall that $R_{\mu,\gamma,\chi} = R_{\phi,\theta,\psi} R_{\alpha,\beta}$ and applying $\beta \equiv 0$ then $\mathrm{s}_\mu \mathrm{c}_\gamma = \mathrm{s}_\phi$ and we have

$$\mathrm{s}_\mu = \frac{\mathrm{s}_\phi}{\mathrm{c}_\gamma} \tag{52.22}$$

for normal flight conditions where $\mathrm{c}_\gamma \neq 0$ and $\mathrm{s}_\phi < \mathrm{c}_\gamma$. By simple geometry,

$$\mathrm{c}_\mu = \frac{1}{\mathrm{c}_\gamma} \sqrt{\mathrm{c}_\gamma^2 - \mathrm{s}_\phi^2} \,,$$

and the dependence on the bank angle can be removed entirely from kinematics of $(|v_\mathrm{a}|, \gamma, \chi)$ to be replaced by the roll angle ϕ.

Avidbots Ex1016
Page 1339 of 2154

Inverting (52.21) and substituting for (52.2b), (52.19) and for (52.22) we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}|v_\mathrm{a}| = -g\mathrm{s}_\gamma + \frac{T}{m}\mathrm{c}_\alpha - \alpha\frac{\bar{Q}S}{m}(C_\mathrm{D}^\alpha + C_\mathrm{L}) - \frac{\bar{Q}S}{m}C_\mathrm{D}\,, \tag{52.23a}$$

$$\dot{\chi} = -\frac{\mathrm{s}_\phi}{\mathrm{c}_\gamma}\left(\frac{\bar{Q}S}{mc_\gamma}C_\mathrm{L} + (C_\mathrm{L}^\alpha - C_\mathrm{D})\alpha\right), \tag{52.23b}$$

$$\dot{\gamma} = \frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|} + \alpha\frac{\bar{Q}S}{m}(C_\mathrm{L}^\alpha - C_\mathrm{D}) - C_\mathrm{L}\frac{\bar{Q}S\sqrt{\mathrm{c}_\gamma^2 - \mathrm{s}_\phi^2}}{m|v_\mathrm{a}|\mathrm{c}_\gamma}\,, \tag{52.23c}$$

where one should recall that $\bar{Q} = \frac{1}{2}\rho|v_\mathrm{a}|^2$ (52.13) depends on $|v_\mathrm{a}|$.

Note that for α small, T provides a free input to stabilize the speed of the vehicle in (52.23a), although we have seen in Sect. 52.3.2 that a different approach is advantageous in practice. In (52.23b), there is no direct input variable, however, the roll ϕ will play this role in the control design. Similarly, in (52.23c), the angle-of-attack α, which itself has dynamics, must be used to control γ. In fact, both (52.23b) and (52.23c) depend on a complex mix of variables including dynamic pressure (and consequently $|v_\mathrm{a}|$) as well as ϕ, α and γ. Nevertheless, the identification of roll ϕ with control of heading and angle-of-attack α with control of flight path is a natural approach in the development of aircraft control.

The kinematics of ϕ are straightforward to compute by differentiating (52.1). Differentiating the $(3,2)$ entry of $R_{\phi,\theta,\psi}$ (52.1) we have

$$\dot{\phi} = \Omega_x\mathrm{c}_\theta + \Omega_z\mathrm{s}_\theta\,. \tag{52.24}$$

The primary control for $\dot{\phi}$ is the roll angular velocity Ω_x and this control will be used to cancel the disturbance created by Ω_z as discussed in Sect. 52.3.2.

To model the dynamics of α, recall the relationship $R_{\mu,\gamma,\chi} = R_{\phi,\theta,\psi}R_{\alpha,\beta}$ again. This time considering the $(3,1)$ element of $R_{\mu,\gamma,\chi}$ we obtain

$$\begin{aligned}\dot{\alpha}\,(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\alpha\mathrm{c}_\phi\mathrm{c}_\theta) = {}& -\mathrm{s}_\phi\mathrm{c}_\alpha\Omega_x + \mathrm{s}_\phi\mathrm{s}_\alpha\Omega_z \\ & + (\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\phi\mathrm{c}_\theta\mathrm{c}_\alpha)\Omega_y \\ & - \mathrm{c}_\gamma\dot{\gamma}\,.\end{aligned} \tag{52.25}$$

Back substituting from (52.23c) and rearranging yields

$$\dot{\alpha} = \Omega_y + \frac{\mathrm{s}_\phi\mathrm{s}_\alpha\Omega_z - \mathrm{s}_\phi\mathrm{c}_\alpha\Omega_x}{(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\alpha\mathrm{c}_\phi\mathrm{c}_\theta)} - \mathrm{c}_\gamma\left[\frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|} + \alpha\frac{\bar{Q}S}{m}(C_\mathrm{L}^\alpha - C_\mathrm{D}) - \frac{\bar{Q}Sc_\mu}{m|v_\mathrm{a}|}C_\mathrm{L}\right], \tag{52.26}$$

where the denominators are always well defined for normal flight conditions. The angular velocity Ω_y enters (52.26) as a free input providing the control for AOA.

Equations (52.23a), (52.23c) and (52.26) lead to a cascade nonlinear system

$$\dot{\chi} = -\mathrm{s}_\phi\left(\frac{|v_\mathrm{a}|^2}{\mathrm{c}_\gamma^2}A_2 + \frac{\alpha}{\mathrm{c}_\gamma}A_3\right), \tag{52.27a}$$

$$\dot{\phi} = u_\phi\,, \tag{52.27b}$$

$$\dot{h} = -|v_\mathrm{a}|\mathrm{s}_\gamma\,, \tag{52.27c}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}|v_\mathrm{a}| = TA_1 - g\mathrm{s}_\gamma + f_1(\gamma,\alpha,|v_\mathrm{a}|)\,, \tag{52.27d}$$

$$\dot{\gamma} = \alpha|v_\mathrm{a}|^2A_4 - \frac{|v_\mathrm{a}|}{\mathrm{c}_\gamma}(\mathrm{c}_\gamma^2 - \mathrm{s}_\phi^2)^{\frac{1}{2}}A_5 + f_2(\gamma,|v_\mathrm{a}|)\,, \tag{52.27e}$$

$$\dot{\alpha} = u_\alpha + f_3(\gamma,\alpha,|v_\mathrm{a}|;t)\,, \tag{52.27f}$$

with the following constants

$$A_1 = \frac{\mathrm{c}_\alpha}{m}\,,\quad A_2 = \frac{C_\mathrm{L}\rho S}{2m}\,,\quad A_3 = (C_\mathrm{L}^\alpha - C_\mathrm{D})\,,$$
$$A_4 = \frac{\rho S}{2m}(C_\mathrm{L}^\alpha - C_\mathrm{D})\,,\quad A_5 = C_\mathrm{L}\frac{\rho S}{2m}\,,$$

functions;

$$f_1(\gamma,\alpha,|v_\mathrm{a}|) = -\alpha\frac{\bar{Q}S}{m}(C_\mathrm{D}^\alpha + C_\mathrm{L}) - \frac{\bar{Q}S}{m}C_\mathrm{D}\,,$$
$$f_2(\gamma,|v_\mathrm{a}|) = \frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|}\,,$$
$$f_3(\gamma,\alpha,|v_\mathrm{a}|) = -\mathrm{c}_\gamma\left[\frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|} + \alpha\frac{\bar{Q}S}{m}(C_\mathrm{L}^\alpha - C_\mathrm{D}) - \frac{\bar{Q}Sc_\mu}{m|v_\mathrm{a}|}C_\mathrm{L}\right],$$

and the input functions;

$$u_\phi = \Omega_x\mathrm{c}_\theta + \Omega_z\mathrm{s}_\theta\,,$$
$$u_\alpha = \Omega_y + \frac{\mathrm{s}_\phi\mathrm{s}_\alpha\Omega_z - \mathrm{s}_\phi\mathrm{c}_\alpha\Omega_x}{(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\alpha\mathrm{c}_\phi\mathrm{c}_\theta)}.$$

Here the height h kinematics have been added to complete the bank-to-turn dynamics. The course (52.27a), and the height (52.27c) dynamics have negative signs compared to conventional fixed-wing modeling developments due to the nonstandard north-west-up convention chosen.

These equations are fully nonlinear equations based on the sliding mode simplification associated with holding $\beta \equiv 0$. The condition on side-slip angle will can be enforced using the free degree of freedom in Ω_z as will be discussed in Sect. 52.3.2. We note that the dynamics presented in this section are closely related to the *longitudinal dynamics* considered in classical aeronautical texts that treat the angle of attack, glide angle, and velocity of the vehicle. The more classical expressions for longitudinal control of a fixed-wing vehicle are obtained from the above model by setting $\dot{\chi} = \mu = \phi = 0$. An additional simplifying assumption often made for the course kinematics (52.27a) is that lift is compensating gravity during level-flight. With these assumptions, (52.27a) can be replaced by

$$\dot{\chi} = -\frac{g}{|v_a|}\tan(\phi)$$

based on centripetal force balance.

52.3 Control

Control for aerial vehicles is challenging for a number of reasons. First, most aerial vehicles are under actuated and the control design must exploit interactions between dynamic states to control the vehicle. Second, aerial vehicles use aerodynamic effects for thrust and lift generation, and regulation of these forces is by nature approximate, leading to significant modeling error. Third, external effects such as wind, turbulence, and vortex generation, lead to high levels of load disturbance in the control loops. Finally, it is often difficult or impossible to measure the vehicle and aerodynamic state directly making it necessary to use an observer or design the controller from first principles to use explicit measurements.

Most aerial vehicles are controlled using a hierarchical control structure with nested feedback loops based on three levels; planning, guidance, and control:

- *Planning*: It is the outermost loop in aerial vehicle control and is associated with path planning, setting way-points, etc.
- *Guidance*: The guidance level of control concerns tracking trajectories to achieve local goals. This control loop is typically designed using the attitude reference as a virtual input, leading to a fully actuated control problem.
- *Control*: It is the high gain innermost loop of the control hierarchy and is concerned primarily with attitude and flight stability of the vehicle. This control problem is typically fully actuated and can be tackled using standard control techniques.

The more dynamic and aggressive the task considered is, the more these levels of control will interact and an integrated control design must be considered. However, in most real-world situations, splitting the control into three hierarchical levels leads to a simpler design problem that achieves the desired performance. For example, the recent aggressive maneuvers undertaken and demonstrated in quadrotor control are based on hierarchical control strategies that use trajectory planning processes for the outer loop that specify achievable trajectories that then are stabilized in real-time using attitude as a virtual input, with high-gain feedback on the inner-loop attitude stability control. Most of this work has been done with the use of external motion capture systems to provide the inertial frame position and velocity feedback [52.7–12, 32]. However, more recently there have been a few attempts to do this with onboard cameras and inertial measurement units [52.13, 14].

52.3.1 Quadrotor Control

The quadrotor control and guidance problem is conceptually more straightforward than that of fixed-wing vehicles. In its simplest form, the goal is to design control algorithms to track smooth feasible trajectories $(R^\star(t), \xi^\star(t)) \in \mathrm{SE}(3)$. We will assume that a planner specifies the full desired trajectory, including the higher order derivative terms $(\Omega^\star(t), \dot{\xi}^\star(t))$ and $(\dot{\Omega}^\star(t), \ddot{\xi}^\star(t))$. A quadrotor vehicle is an under-actuated system – there are four inputs $\mathbf{u} = (T, \tau^{\mathrm{T}})^{\mathrm{T}}$, while the trajectory lives in SE(3) and is six-dimensional. The hierarchical control structure proposed leads to an inner loop regulating the attitude using the moments τ as the control input. The guidance level utilizes the attitude R and heave T to regulate the trajectory $\xi(t)$ to track $\xi^\star(t)$. In high performance control of a quadrotor, the major limitation to performance comes from the limits of motor response and the control hierarchy must be augmented by a low level motor regulation system. The proposed control architecture forms nested feedback loops as shown in Fig. 52.6.

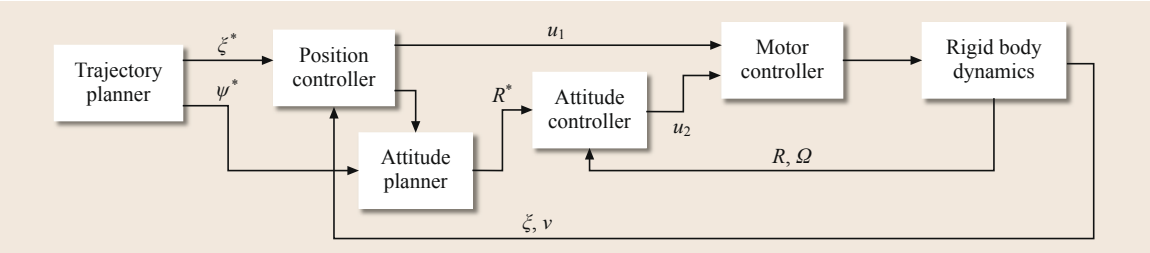


Fig. 52.6 Typical control architecture. A base-level motor regulation, an inner attitude control loop, an intermediate position guidance loop, with an outer trajectory planner

Motor Regulation

The aerodynamics of thrust generation by rotors is a subject that has been studied in detail in the classical rotorcraft literature [52.22, 23, 27]. Using a full model of rotor aerodynamics has significant potential in the design of the motor control for high-performance rotor control for aerial robotic applications [52.33]. Despite this, it is sufficient in most cases to consider a static thrust model that holds for a rotor in hover and is a reasonable approximation for most robotic applications. In such a model, the thrust generated is proportional to the square of the rotor speed. Since most quadrotor vehicles are equipped with brushless DC motors that use back EMF sensing for rotor commutation, making it possible to measure rotor angular velocity, the control of the rotor thrust is usually implemented as a local control loop on the electronic speed controller (ESC) regulating rotor speed.

Most ESC use PWM (pulse width modulation) regulation of voltage bus to provide voltage control of the motors. A typical rise time for the uncontrolled system is in the order of 200 ms and it is necessary to include a local control loop to improve the system response. The aerodynamic drag of the rotor means that the system is naturally heavily damped and there is no need for derivative control. Similarly, an integral term is rarely deemed necessary at the rotor control level since the thrust model used is not particularly accurate anyway and it will always be necessary to have integral control at higher levels in the control hierarchy. Since an integral term is inadvisable, it is important to use the best model of the thrust generation possible as a feedforward term $V_{\mathrm{ff}}(\varpi_i^\star)$ to minimize the requirement on the proportional regulation. Thus, a typical ESC rotor control is given by saturated proportional control with feedforward [52.34]

$$V_i = \mathrm{sat}\left[k(\varpi_i^\star - \varpi_i) + V_{\mathrm{ff}}(\varpi_i^\star)\right], \tag{52.28}$$

where V_i is the applied motor voltage, $\varpi_i^\star$ is the desired speed and the actual motor speed ϖ_i is measured from the electronic commutation in the embedded speed controller. The performance of the motor controllers is ultimately limited by the current that can be supplied from the batteries [52.34] and a saturation on the demanded voltage is necessary. Without the saturation, extreme maneuvers may cause the voltage bus to drop excessively, destroying the rotor regulation performance and unless care is taken, causing the onboard electronics to brownout.

Attitude Control

The control problem considered in attitude regulation is to use full actuation of τ in (52.2d) to control (52.2c) to track a desired attitude trajectory $R^\star(t)$ along with its velocity $\Omega^\star(t)$ such that $\dot{R}^\star(t) = R^\star(t)\Omega^\star(t)$. The approach described uses a global stability design based on minimizing the matrix error

$$\tilde{R} = (R^\star)^{\mathrm{T}} R \tag{52.29}$$

similar to [52.7, 35]. Driving $\tilde{R} \to I_3$ ensures that $R \to R^\star$. The kinematics of the tracking error is given by

$$\begin{aligned}\dot{\tilde{R}} &= -\Omega_\times^\star \tilde{R} + \tilde{R}\Omega_\times \\ &= [\tilde{R}, \Omega_\times^\star] + \tilde{R}(\Omega_\times - \Omega_\times^\star) \\ &= [\tilde{R}, \Omega_\times^\star] + \tilde{R}\tilde{\Omega}_\times \,,\end{aligned} \tag{52.30}$$

where $[A, B] = AB - BA$ is the matrix commutator and

$$\tilde{\Omega} := \Omega - \Omega^\star \,. \tag{52.31}$$

We assume that derivative $\dot{\Omega}^\star$ of the target angular velocity is available for the purposes of designing a feedforward torque for the angular velocity dynamics

$$\tau^\star := \mathbf{I}\dot{\Omega}^\star + \Omega_\times^\star \mathbf{I}\Omega \,, \tag{52.32}$$

where $\mathbf{I}$ is the inertia matrix of the airframe (52.2d). Feed-forward control for the angular velocity of the attitude is important for high-performance maneuvers, especially when the motor response is pushed to near performance limits. The higher derivative information

Avidbots Ex1016
Page 1342 of 2154

for the angular velocity is available from path planning algorithms such as those discussed in Sect. 52.4.1 or using predictive control algorithms [52.8–10, 36]. Choose the control input

$$\tau := \tau^\star + \mathbf{u}_2 \tag{52.33}$$

where $\mathbf{u}_2 \in \mathbb{R}^3$ is the free control that will be used to stabilize the error dynamics (Fig. 52.6). The error dynamics for the angular velocity error are now

$$\mathbf{I}\dot{\tilde{\Omega}} = -\tilde{\Omega}_\times \mathbf{I}\Omega + \mathbf{u}_2 \,. \tag{52.34}$$

The goal is to choose $\mathbf{u}_2$ in order to stabilize the error dynamics (52.30) and (52.34) robustly. Define

$$\mathbb{P}(\tilde{R}) := \frac{1}{2}\left(\tilde{R} - \tilde{R}^{\mathrm{T}}\right)$$

to be the skew-symmetric projection of the error matrix. It can be verified that $\mathbb{P}(\tilde{R}) = \sin(\theta) a_\times$ where (a, θ) is the angle-axis representation of $\tilde{R}$. That is the skew-symmetric projection is the axis of rotation that rotates R to $R^\star$ (equivalently rotates $\tilde{R}$ to the identity) scaled by the sin of the angle between the two orientations. Choose a proportional–derivative (PD) control

$$\mathbf{u}_2 = -k_{\mathrm{P}} \mathrm{vex}(\mathbb{P}(\tilde{R})) - k_{\mathrm{D}} \tilde{\Omega} \,,$$

where vex: $\mathbb{R}^{3\times 3} \to \mathbb{R}^3$ is the inverse of the skew-symmetric operator applied to skew-symmetric matrices, $\mathrm{vex}(\Omega_\times) = \Omega$. The proportional term applies a torque associated with a nonlinear spring associated with a potential energy $\mathrm{tr}(\tilde{R}^{\mathrm{T}}\tilde{R})$ while the derivative $k_{\mathrm{D}}\tilde{\Omega}$ provides damping. It is straightforward to demonstrate that the Lyapunov function

$$\mathcal{L} := k_{\mathrm{P}} \mathrm{tr}(\tilde{R}^{\mathrm{T}}\tilde{R}) + \frac{1}{2}\tilde{\Omega}^{\mathrm{T}}\mathbf{I}\tilde{\Omega}$$

is decreasing

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L} := -k_{\mathrm{D}} \|\tilde{\Omega}\|^2 \,.$$

Invoking Barbalat's lemma, with some care, it follows that the system is almost globally asymptotically stable. The zero measure exception set for the basin of attraction is associated with situations where the quadrotor is completely flipped, $\theta = \pi$ rad, and $\mathbb{P}(\tilde{R}) = 0$ even though $\tilde{R} \neq I$.

To verify the local exponential convergence and provide guidance for the gain tuning, we consider the linearization of (52.30) and (52.34). Write $\tilde{R} \approx I + (z_R)_\times$ where the skew-symmetric matrix $(z_R)_\times$ (for $z_R \in \mathbb{R}^3$) is the linear approximation of $\tilde{R}$ around the identity matrix I. Write $\tilde{\Omega} \approx z_\Omega$ for the linear approximation of $\tilde{\Omega}$ around the origin $\tilde{\Omega} = 0$. The linearization of the error $z = (z_R, z_\Omega)$ system is given by

$$\begin{pmatrix} \dot{z}_R \\ \dot{z}_\Omega \end{pmatrix} = \left[\begin{pmatrix} (\Omega^\star)_\times & 0 \\ 0 & \mathbf{I}^{-1}(\mathbf{I}\Omega)_\times \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ -k_{\mathrm{P}}\mathbf{I}^{-1} & -k_{\mathrm{D}}\mathbf{I}^{-1} \end{pmatrix} \right] \begin{pmatrix} z_R \\ z_\Omega \end{pmatrix} \tag{52.35}$$

or in more compact notation $\dot{z} = A(t)z$ where $A(t) = A_1(t) + A_2$ comprises the sum of two matrices in (52.35). The matrix $A_1(t)$ is a time-varying matrix depending on the exogenous system signals Ω and $\Omega^\star$ while the second matrix A_2 is a time-invariant Hurwitz second-order linear system matrix. It is straightforward to show that this system is asymptotically stable using a Lyapunov function $\ell = k_{\mathrm{P}}/2|z_R|^2 + 1/2 z_\Omega^{\mathrm{T}}\mathbf{I}z_\Omega$, using an analog of the nonlinear argument made earlier. We have that $\dot{\ell} = -k_{\mathrm{D}}|z_\Omega|^2$. Define $C = \begin{pmatrix} 0 & I_3 \end{pmatrix} \in \mathbb{R}^{3\times 6}$ then if the pair $(A(t), C)$ is uniformly completely observable (UCO) asymptotic stability implies exponentially stability [52.37, pp. 626–628]. It is intuitively clear that the system matrices $(A(t), C)$ associated with (52.35) are UCO, although providing the algebraic proof is beyond the scope of the this chapter. Tuning the gains is then a process of choosing k_{P} and k_{D} to assign the eigenvalues of the Hurwitz system matrix, with an eye to dominate the oscillatory, but bounded energy, disturbance introduced by $A_1(t)$.

Trajectory Tracking Control

The trajectory planning algorithm, discussed later in Sect. 52.4.1, provides a full trajectory specification as well as the feedforward inputs consisting of $(\xi^p, \dot{\xi}^p, R^p, \Omega^p, \dot{\Omega}^p, u_1^p, u_2^p)$ and their derivatives. The goal of trajectory tracking control is to regulate the linear dynamics of a quadrotor to track a specified goal trajectory $\xi^\star(t)$. Trajectory tracking control operates at a higher level in the hierarchy of the quadrotor control architecture (Fig. 52.6) and the desired attitude $R^\star$ used in the attitude control loop is designed as an input in the trajectory loop control design. In addition to specifying the attitude goal, $R^\star$, the trajectory tracking control specifies the heave (total thrust), T, input reference that is used along with (52.33) for the low-level motor control reference (52.9).

The linear trajectory goal $\xi^\star = \xi^p$ and $\dot{\xi}^\star = \dot{\xi}^p$ are used directly from the path planning design as the goal of the trajectory planner. The actuation for the linear dynamics, however, depends on the attitude of the vehicle. Thus, it is necessary to define the goal for the attitude control $R^\star$ as a part of the control design

rather than rely on the planned trajectory set-point; in particular $R^\star \neq R^p$ in general. Note that the feedforward control in Sect. 52.3.1 requires $\Omega^\star$ and $\dot{\Omega}^\star$ to plan the feedforward torque input for the attitude dynamics. Feed-forward in the attitude regulation loop is a critical component of high-performance control and without this input, there will not be sufficient gain in the attitude dynamics to track aggressive attitude trajectories. However, computing the angular velocity goal that corresponds to a feedback dependent attitude control goal $R^\star$ in real-time requires forward prediction of the control trajectory using tools such as model predictive control (MPC) [52.8–10, 36]1. A simpler approach for trajectory tracking is to use the planned angular velocity $\Omega^\star = \Omega^p$ even though it does not exactly match the variation of $R^\star$. As long as $(\xi, \dot{\xi})$ is close to $(\xi^\star, \dot{\xi}^\star) = (\xi^p, \dot{\xi}^p)$ then $R^\star$ is close to R^p and $(\Omega^\star, \dot{\Omega}^\star) = (\Omega^p, \dot{\Omega}^p)$ will provide a good estimate of the critical feedforward attitude input.

We will discuss here the linear trajectory tracking algorithm for which the specified trajectory is required to have roll and pitch values that are small thus justifying a linear approximation of the dynamics. Accordingly, we assume that the Euler angles associated with the rotation R^p (52.1), are given by $\theta^p = \phi^p = 0$, while ψ^p is a specified function of t. For a given arbitrary yaw angle, we can linearize the dynamics about the hover position ($\theta = 0$, $\phi = 0$, $\psi = \psi^p$) and the nominal inputs $u_1 = mg$, $\mathbf{u}_2 = 0$. Linearizing (52.2a), we obtain

$$\begin{aligned}
\ddot{\xi}_1 &= g(\Delta\theta \cos\psi^p + \Delta\phi \sin\psi^p) \,, \\
\ddot{\xi}_2 &= g(\Delta\theta \sin\psi^p - \Delta\phi \cos\psi^p) \,, \\
\ddot{\xi}_3 &= \frac{1}{m} u_1 - g \,,
\end{aligned} \tag{52.36}$$

where $\Delta\theta = \theta^\star$ and $\Delta\phi = \phi^\star$ represent the small deviations from the hover position. In order to exponentially drive all three components of error in the position part of the trajectory, we want the acceleration vector $\ddot{\xi}$ to satisfy

$$(\ddot{\xi}^\star - \ddot{\xi}) + K_d(\dot{\xi}^\star - \dot{\xi}) + K_p(\xi^\star - \xi) = 0 \,.$$

From (52.36), we can immediately write

$$u_1 = m\left[g + \ddot{\xi}_3^\star + k_{d,z}(\dot{\xi}_3^\star - \dot{\xi}_3) + k_{p,z}(\xi_3^\star - \xi_3)\right], \tag{52.37}$$

to guarantee $(\xi_3(t) - \xi_3^\star(t)) \to 0$. We choose the appropriate desired roll, pitch, and yaw angles for $\theta^\star$, $\phi^\star$, and $\psi^\star$ to guarantee exponential convergence:

$$\phi^\star = \frac{1}{g}(\ddot{\xi}_1^\star \sin\psi - \ddot{\xi}_2^\star \cos\psi) \,, \tag{52.38a}$$

$$\theta^\star = \frac{1}{g}(\ddot{\xi}_1^\star \cos\psi + \ddot{\xi}_2^\star \sin\psi) \,, \tag{52.38b}$$

$$\psi^\star = \psi^p \,. \tag{52.38c}$$

Now $(\psi^\star, \phi^\star, \theta^\star)$ define the rotation matrix $R^\star$ (52.1) provided as a set point for the exponentially convergent attitude controller discussed in the previous section. Thus, as shown in Fig. 52.6, the control problem is addressed by decoupling the position control and attitude control subproblems and the position control loop provides the attitude set points for the attitude controller.

The position controller can also be obtained without linearization. This is done by projecting the position error (and its derivatives) along $\mathbf{b}_3$ and applying the input u_1 that cancels the gravitational force and provides the appropriate proportional plus derivative feedback

$$u_1 = m\boldsymbol{b}_3^{\mathrm{T}}\left[\ddot{\xi}^\star + K_d(\dot{\xi}^\star - \dot{\xi}) + K_p(\xi^\star - \xi) + ge_3\right]. \tag{52.39}$$

Note that the projection operation is a nonlinear function of the roll and pitch angles and thus this is a nonlinear controller. In [52.7, 35], it is shown that this approach results in exponential stability and allows the robot to track trajectories in SE(3).

52.3.2 Control of Fixed-Wing Aircraft

An aircraft is a highly coupled nonlinear dynamical system and it may seem most natural to treat the control problem as an integrated multi-input multi-output (MIMO) nonlinear design [52.29, 38–40]. However, despite the nonlinear coupling of aircraft states there are still clear dependencies between specific actuators and certain dynamic modes of the open-loop response. A disadvantage of the MIMO approach is that it tends to obscure insight between control design and specific open-loop mode response. Moreover, and as a consequence, it tends to be less robust to large changes in model parameters (such as caused by using a different set of wings, or bolting on a completely new sensor package, not an uncommon situation for a small UAV system) than a classical control design. The classical, and more intuitive, control architecture for fixed wing vehicles is to consider a set of separate, but interconnected, single-input single-output (SISO) control loops. Such an approach has the advantage of being highly intuitive and leading to a modular design methodology

Avidbots Ex1016
Page 1344 of 2154

based on successive loop closure, providing a straightforward gain tuning regime [52.31]. The robustness and simplicity of the SISO architecture make it the preferred choice for most small UAV control systems. A modern implementation, however, uses the total energy control system (TECS) architecture and input decoupling to overcome some of the limitations of classical fixed-wing control design.

The main control actions for a fixed wing aircraft are associated with the throttle, and the three aerodynamic control surfaces; rudder, ailerons, and elevators. The throttle regulates power to the motor and governs thrust T. The rudder, ailerons and elevators actuate the angular velocities Ω_z, Ω_x, and Ω_y, respectively, through the angular dynamics of the vehicle. For a small fixed-wing UAV in normal flight, a simple high-gain proportional control scheme will effectively regulate and decouple the variables $(T, \Omega_x, \Omega_y, \Omega_z)$ [52.31]. The control problem can then be posed as one where $(T, \Omega_x, \Omega_y, \Omega_z)$ are inputs for regulation of the remainder of the vehicle dynamics. Since there is weak physical coupling between the lateral variables (roll rate, bank angle, yaw rate, and course angle) and the longitudinal variables (airspeed, pitch rate, pitch angle, altitude rate, and altitude), and since the elevator and rudder most strongly effect the lateral variables and the throttle and aileron most strongly influence the longitudinal variables, most control architectures separate functionality into a lateral control system and a longitudinal control system. The input to the lateral control system is the desired course angle, and the input to the longitudinal control system is the desired airspeed and the desired altitude.

Lateral Control System

There are three open-loop dynamic modes associated with the response of the lateral dynamics of a fixed-wing aircraft: the roll mode, the spiral mode, and the dutch roll mode. The *roll mode* is associated with the first-order response of the roll rate of the vehicle to aileron input. Usually the wings and airframe are designed (e.g., with a positive dihedral angle Chap. 26.4.4) so that this mode is highly stable and is regulated using high-gain in the aileron input – indeed, this mode is subsumed into the control simplification (regulation of Ω_x) proposed above.

The spiral mode is associated with the natural turning motion of the vehicle. That is, if the vehicle is banked into a turn, it will continue to turn until control action is applied to flatten the vehicle out. The spiral mode is a linearization of the bank-to-turn dynamics and is a single real pole close to the origin (Chap. 26, Fig. 26.22). Typically the spiral mode is slightly unstable due to the tendency of the vehicle to side-slip into a turn. In manned flight, the spiral mode can be dangerous since there is very little external perception of turning. The nature of the banked turn dynamics ensure that the body-fixed-frame acceleration remains oriented through the central axis of the vehicle and rate of rotation is slow and nearly constant. The mode becomes dangerous if the pilot is not aware as the vehicle slowly side-slips down into the turn, progressively diving more and more steeply and increasing airspeed in what is known as a *spiral-dive*. For autonomous flight control, the spiral mode is relatively benign since autopilots never stop looking at the instruments and take immediate action if the vehicle is drifting off course.

The *dutch-roll* mode, is associated with the oscillation of the vehicle around a heading. In this motion, the vehicle yaws away from the heading and then rolls due to the dihedral effect, before the tilted lift vector and rudder action force the airplane back toward the incident wind. If the rudder is small and does not provide sufficient damping, the vehicle can overshoot to yaw and roll in the other orientation before repeating the oscillatory sequence. Dutch role motion is less dangerous for manned flight since it is usually stable (except for vehicles with strongly swept back wings or negative dihedral) and produces unpleasant sideways acceleration with the side-slip motion that is immediately apparent to a pilot. Damping is increased by large tail plane surfaces and moderate to small dihedral. A figure of the typical pole positions of an aerobatic RC airplane is shown in Chap. 26, Fig. 26.22.

To understand the control problem for lateral dynamics, consider first the angular yaw velocity Ω_z induced by rudder actuation. The primary effect of Ω_z will yaw the vehicle in the air and generate nonzero side-slip β. In the *bank-to-turn* control paradigm described later, we will use Ω_z to regulate $\beta \equiv 0$, however, it is instructive first to consider the effect of nonzero side-slip in order to build an intuitive understanding of the lateral dynamics. If the side-slip $\beta \neq 0$, the airplane will be sliding sideways or crabbing through the air. In particular, applying rudder input (nonzero Ω_z) acts to twist the airplane in the air; it does not in itself cause the trajectory of the airplane to curve. For a typical airplane with dihedral, this will lead the aircraft to bank away from the side-slip, and if the dutch-roll mode is stable, the airplane will settle into a banked turn. The causality of the system signals is given by

$$\Omega_z \rightarrow \beta \rightarrow \phi \rightarrow \chi \, .$$

Here the actuation from $\beta \rightarrow \phi$ relies on the inherent stability of the dutch roll mode of the lateral open-loop dynamics of the airframe. Indeed, small-scale vehicles with strong dihedral and large tail planes can be flown

Avidbots Ex1016
Page 1345 of 2154

without aileron surfaces. For example, the aerial robotic vehicle as shown in Fig. 52.7, designed for slow low-level flight and equipped for visual terrain tracking using optic flow computed from a pair of downward pointing web cameras [52.41], does not have ailerons and the lateral motion is controlled using only the rudder.

This control paradigm has several negatives. It is rarely used for manned flight because side-slip motion generates sideways accelerations that do not feel natural. More importantly, the control response is limited by the low pass response of the open loop dutch roll dynamics of the aircraft. Although this mode is stable, there is often a noticeable damped oscillatory response that causes the vehicle trajectory to wander around the desired heading and degrades course tracking performance.

The alternative is to explicitly use the ailerons to impose the desired roll angle ϕ on the vehicle and actuate the heading dynamics (52.27a) without using the rudder. The rudder then becomes a separate control input that can be used to regulate β. That is

$$\Omega_z \to \beta \,,$$
$$\Omega_x \to \phi \to \chi \,.$$

Although, this may seem counter intuitive, (the rudder is not used to steer the airplane) it is, however, the most effective heading control strategy. Using this control strategy and regulating the side-slip $\beta \equiv 0$ ensures that the bank-to-turn simplifications can be applied to the model as was shown in Sect. 52.2.3.

Specifically, in the bank-to-turn strategy where side-slip β is regulated to zero using the rudder, the lateral equations of motion are given by (52.24) and (52.27a). In level flight where the pitch angle is small and the velocity and angle of attack are roughly a constant, these equations simplify to the cascade structure

$$\dot{\phi} = \Omega_x + d_\phi \,,$$
$$\dot{\chi} = -A_\chi \phi + d_\chi \,,$$

where A_χ is a constant and d_ϕ and d_χ are disturbance signals introduced through the modeling process. From classical control theory, we know that a step in the heading command $\chi^\star$ can be tracked while simultaneously rejecting low-frequency disturbances d_ϕ and d_χ using a nested PI control law given by

$$\phi^\star = k_{\mathrm{P}_\chi}(\chi^\star - \chi) + k_{I_\chi} \int_{-\infty}^{t} (\chi^\star - \chi)\,\mathrm{d}\tau \,, \quad (52.40)$$

$$\Omega_x = k_{\mathrm{P}_\phi}(\phi^\star - \phi) + k_{I_\phi} \int_{-\infty}^{t} (\phi^\star - \phi)\,\mathrm{d}\tau \,, \quad (52.41)$$

where k_{P_χ}, k_{I_χ}, k_{P_ϕ}, and k_{I_ϕ} are positive control gains. The proportional control

$$\Omega_z = -k_{\mathrm{P}_\beta}\beta, \quad (52.42)$$

is usually effective at regulating the side-slip angle to zero, where k_{P_β} is a positive control gain.



Fig. 52.7 An autonomous fixed-wing aerial robot for experimental work on vision-based terrain tracking at ANU (after [52.41]). Note the upswept wings to generate dihedral and the lack of ailerons

Longitudinal Control System

The longitudinal dynamics concern the thrust T and elevator (regulating Ω_y) inputs, along with the angle of attack α, the flight path angle γ, the airspeed $|v_\mathrm{a}|$, and the altitude h. There are two primary open-loop dynamic modes associated with the open loop response of the longitudinal dynamics, the short-period and phugoid modes. The *short-period* mode is associated with variation of pitch of the vehicle for constant velocity, constant altitude flight. It is a consequence of the linear dependence of lift of the primary wing to angle-of-attack as the vehicle pitches up and down around its center of the mass. The center of mass of an airplane is always designed to lie in front of the center of force of the primary wing surface to ensure that the short-period mode oscillation is stable, (placing the mass further forward increases the stability margin,) while damping provided by the tail plane of the vehicle ensures asymptotic stability (Chap. 26.4.2).

The *phugoid mode* is associated with the interplay of aerodynamic lift, velocity, glide path, and height of the vehicle for constant angle-of-attack flight. As a vehicle starts to descend gently, the airspeed increases as the loss of potential energy in height is transferred into kinetic energy. As the vehicle accelerates, the increased airspeed leads to increased primary lift on the wings leading the vehicle to pull out of the dive and to start to ascend again. The vehicle will slow as the

Avidbots Ex1016
Page 1346 of 2154

stored kinetic energy is traded back into potential energy. The resulting open-loop unstabilized motion is a repeated swooping cycle. Since aircraft are designed with low drag coefficients, the phugoid mode is usually very lightly damped, however, it is slow and easily stabilized. A key point to note in phugoid motion is that it is a constant energy oscillation; potential energy is transformed into kinetic energy and is then transformed back into potential energy.

Both thrust and pitch rate inputs affect both velocity and angle-of-attack of an airplane. In turn, the glide-path angle can be controlled through regulation of either angle-of-attack or velocity of the vehicle. The coupling associated with these dependencies mean that simple thrust to velocity or pitch-rate to glide-path SISO control loops cannot cover the full flight regime of a typical aircraft, a limitation of many existing autopilot systems even in commercial airliners. The industry standard approach to address this issue is to have different control modes for different flight conditions, take-off and climb, level flight, and descent, that exploit different input to state control mappings [52.31]. The total energy control system (TECS) design paradigm [52.42, 43] partially avoids these difficulties for a wide range of operating conditions and has the additional advantage of minimizing the control activity undertaken by the throttle.

Due to the energy preserving nature of the phugoid mode, small variations in velocity and height tend to be stable and energy conserving. Based on this insight, it is desirable to allow small variations of the airspeed of the vehicle as long as it is associated with the phugoid mode response. The key characteristic of the phugoid mode is that it preserve *total energy*

$$E := \frac{1}{2}m|v_a|^2 + mgh \tag{52.43}$$

of the aircraft. The TECS control architecture uses the total energy as an output and uses the throttle to regulate this output to a desired energy set point. The set point

$$E^\star(t) := \frac{1}{2}m|v_a^\star|^2 + mgh^\star$$

is chosen to correspond to the desired trajectory. The energy error is

$$\tilde{E} := E^\star(t) - E(t)\,,$$

and the throttle input is assigned as

$$\delta_t = k_{P_E}\tilde{E}(t) + k_{I_E}\int_{-\infty}^{t}\tilde{E}(\tau)\,\mathrm{d}\tau\,, \tag{52.44}$$

where k_{P_E} and k_{I_E} are positive control gains.

Once the throttle input is assigned, the remaining longitudinal dynamics must be controlled using the elevator to servo the pitch rate Ω_y. The primary effect of pitch rate is to rotate the wing section of the aircraft, directly effecting the angle-of-attack. Changing the angle-of-attack effects lift and drag generated by the wing, with lift most strongly effected since it depends linearly on α (52.14) while drag is small and only varies with the square of the angle-of-attack (52.15). By definition, the lift L is the component of airfoil force that lies perpendicular to the velocity. As a consequence the lift force L *does not contribute directly to changes in the total energy* E of the vehicle. In particular, actuation of the pitch dynamics of the vehicle is (almost) decoupled from total energy regulation loop. It is now possible to choose an output for the elevator input and have confidence that the resulting SISO loop response will only marginally disturb the total energy regulation loop. Rather than simply regulate the height or velocity, both variables that have been considered as outputs for elevator regulation loops in classical fixed-wing control architectures [52.31], it is of interest to regulate a balance between kinetic and potential energy [52.42].

Consider the kinetic energy and potential energy of the vehicle separately

$$K = \frac{1}{2}m|v_a|^2\,, \quad U := mgh\,.$$

A given flight trajectory corresponding to desired values of $(|v_a^\star(t)|, h^\star(t))$ defines desired set points for kinetic and potential energy $K^\star(t) = \frac{1}{2}m|v_a^\star(t)|^2$ and $U^\star = mgh^\star(t)$. Define the errors in the kinetic and potential energy along the trajectory as

$$\begin{aligned}\tilde{K} &= K^\star(t) - K(t) = \frac{1}{2}m\left[|v_a^\star(t)|^2 - |v_a(t)|^2\right],\\ \tilde{U} &= U^\star(t) - U(t) = mg\left[h^\star(t) - h(t)\right],\end{aligned} \tag{52.45}$$

and define the combined energy balance error $\tilde{B}$ by

$$\tilde{B} = \tilde{K} - \tilde{U}\,.$$

The proposed control strategy is to use the commanded pitch angle $\theta^\star$ to regulate $\tilde{B} \to 0$.

To understand this control architecture, note that if $\tilde{E} = 0$ then $\tilde{K} = \tilde{E} - \tilde{U} = -\tilde{U}$ and hence

$$\tilde{B} = 2\tilde{K} = -2\tilde{U}\,.$$

It follows that driving $\tilde{B}$ to zero forces both $|v_a| \to |v_a^\star|$ and $h \to h^\star$. The real advantage of the balanced energy control architecture occurs when $\tilde{E} \neq 0$. In this case, driving $\tilde{B} \to 0$ will not force either $|v_a|$ or h to its reference set point at all costs, but rather, it will balance the excess or deficit energy between kinetic and potential energy set points around the desired trajectory.

Avidbots Ex1016
Page 1347 of 2154

Balancing the energy error in this way is a highly robust control strategy that maximizes the ability of the throttle to maintain good control of the total energy.

An additional insight is that the difference of the energy terms $\tilde{B}$ is maximally sensitive to excitation in the phugoid mode. In particular, if power flows between potential and kinetic energy levels, then the relative variation in $\tilde{B}$ is maximal compared to any other combined measure incorporating $|v_a|$ and h subject to scaling factors. Thus, the balanced energy control strongly directly damps the phugoid mode that is uncontrolled by the thrust control.

The desired pitch angle is therefore given by

$$\theta^\star = k_{P_B}\tilde{B}(t) + k_{I_B}\int_{-\infty}^{t}\tilde{B}(\tau)\,\mathrm{d}\tau\,, \qquad (52.46)$$

where k_{P_B} and k_{I_B} are positive control gains. In practice, the pitch angle actuates stable angle-of-attack dynamics that in turn lead to regulation of the flight-path angle and finally control the tradeoff encoded in $\tilde{B}$. The stability of the internal dynamics for the proposed SISO control loop are handled by suitable tuning of the PI gains, avoiding the need to explicitly model these dynamics. The effectiveness and robustness of this approach is well established in the literature [52.31].

Control Implementation and Gain Tuning

The lateral control strategy for fixed wing aircraft is based on (52.40), (52.41), and (52.42). In using these commands, we are assuming high-bandwidth feedback loops regulating Ω_x and Ω_z using the ailerons and rudder, respectively. Practical implementation of the lateral control loops required saturating the roll command $\phi^\star$ to $\pm\bar{\phi}$, where, for most airframes, a reasonable value is $\bar{\phi} = 30°$. Commanding higher roll angles usually results in large side-slip angles, which leads to unacceptably large drops in altitude during turns. The roll rate command Ω_x is also saturated to a fraction of the limits of the rate gyro sensor to ensure adequate convergence of the attitude estimator. Practical implementation of (52.40) also requires careful wrapping of the angles χ and $\chi^\star$ around $\pm 180°$ to avoid large heading errors that result in not associating $180°$ with $-180°$.

The gains of the lateral control loops can be tuned in flight by selectively enabling one loop at a time. A strategy that has proven success over numerous flight test is to tune the gains in the following order:

1. Attitude rate loops regulating Ω_x and Ω_z.
2. Side slip gain k_{P_β}.
3. The proportional gain on roll error k_{P_ϕ}, followed by the integral gain k_{I_ϕ}. The integral gain k_{I_ϕ} is often set to zero, since the integrator on the heading loop provides suitable robustness.
4. The proportional gain on the heading loop k_{P_χ}, followed by the integral gain k_{I_χ}.

The use of (52.46) assumes a high-bandwidth pitch control loop using the elevator as an actuator. Practical implementation of (52.44) requires saturating δ_t above by $\delta_t^{\max}$ and below by $\delta_t^{\min} \geq 0$. Saturating the pitch angle is also critical to avoid stall conditions that occur at large angles of attack. An effective method to do this is to saturate the altitude error in computing the error in potential energy, replacing (52.45) with

$$\tilde{U} = mg\,\mathrm{sat}\left[h^\star(t) - h(t), \bar{h}\right], \qquad (52.47)$$

where

$$\mathrm{sat}(x,\ell) = \begin{cases} \ell & \text{if } x \geq \ell\,, \\ -\ell & \text{if } x \leq -\ell\,, \\ x & \text{otherwise}\,. \end{cases}$$

Saturating the altitude error will cause a steady climb rate when $h^\star - h$ is large. To understand why, assume that velocity is well regulated, and that the altitude error is large, then $\tilde{B} \approx -mg\,\mathrm{sat}(h^\star - h, \bar{h}) = -mg\bar{h}$. Ignoring the integrator, this will cause a constant pitch angle command from (52.46). However, simply saturating the pitch angle will not result in good performance since driving

$$\tilde{B} = \tilde{K} - \tilde{U} = \frac{1}{2}m\left(|v_a^\star(t)|^2 - |v_a(t)|^2\right) - mg[h^\star(t) - h(t)]$$

to zero will cause errors in kinetic and potential energy to balance. Therefore, a large unsaturated altitude error will result in the controller increasing the airspeed error. A sustained altitude error will cause the integrator in (52.46) to wind up. Therefore, an integrator antiwindup scheme is essential.

Similar to the lateral control loop, the gains for the longitudinal controller can be tuned in flight by selectively enabling one loop at time. The gains should be tuned in the following order:

1. The pitch attitude loops is first tuned. Pitch attitude is typically controlled using a PID controller. The derivative gain can be tuned to provide adequate damping on the RC controller. The proportional gain is then added and adjusted to provide adequate transient response. The integrator is added afterward to remove steady state error in pitch.
2. The throttle gains k_{P_E} and k_{I_E} in (52.44) are then adjusted. We have found that normalizing the energy error $\tilde{E}$ by the reference kinetic energy $K^{\mathrm{ref}} = \frac{1}{2}m|v_a^{\mathrm{ref}}|$ makes these gains particularly easy to tune,

and that the scaling results in similar gains working across different platforms.

3. The energy balance gains k_{P_B} and k_{I_B} are tuned last. Again, scaling $\tilde{B}$ by K^{ref} seems to simplify the tuning process for different airframes.

An advantage of the TECS is that it reduces the need for different control modes for the longitudinal autopilot. In particular, the *altitude hold mode*, the *climb mode*, and the *descend mode* described in [52.31] are reduced to one mode using TECS. However, we should note that the *take-off mode* described in [52.31] is still needed for this scheme. For small aircraft that are hand launched, or any aircraft whose velocity at take off is significantly below the commanded velocity, the TECS control in (52.46) will cause the vehicle to pitch down to gain airspeed when the airspeed error is large. This pitch-down behavior will cause the airframe to crash if it is executed immediately after take-off. Therefore, the best strategy is to apply full throttle immediately after take-off, and to set the commanded pitch angle to fixed value that is adequate for a climb rate that avoids stall.

52.4 Trajectory Planning

There are many different control tasks that aerial robotic vehicles will be required to undertake in the coming years. In this chapter, it is impossible to cover the whole range of potential tasks along with their associated planning problems. We will consider only basic trajectory or path planning problems for both quadrotor and fixed-wing vehicles. This problem is fundamental to fixed-wing navigation where the distances involved are such that path and trajectory planning is sufficient for almost all goals. For small aerial robotic vehicles flying in cluttered 3-D-spaces, there is a much wider selection of goals including things such as, physical interaction with the environment, obstacle avoidance, etc., as well as simply planning and following a trajectory to a given way-point. However, the trajectory planning provides a basic building block in achieving a wide range of goals, and is a necessary component of achieving high-performance tasks.

52.4.1 Trajectory Planning for Quadrotors

The quadrotor is underactuated and this makes it difficult to plan trajectories in 12-dimensional state space (6DOF position and velocity). However, the problem is considerably simplified if we use the fact that the quadrotor dynamics are differentially flat [52.44]. To see this, we consider the output position $\boldsymbol{\xi}$ and the yaw angle ψ. We show that we can write all state variables and inputs as functions of the outputs (ξ, ψ) and their derivatives. Derivatives of $\boldsymbol{\xi}$ yield the velocity v, and the acceleration,

$$\dot{v} = \frac{1}{m} u_1 \boldsymbol{b}_3 + g e_3 \,.$$

From Fig. 52.3, we see that

$$\boldsymbol{d}_1 = [\cos\psi, \sin\psi, 0]^{\mathrm{T}}$$

and the unit vectors for the body-fixed frame, can be written in terms of the variables ψ and $\dot{v}$ as follows

$$\boldsymbol{b}_3 = \frac{\dot{v} - g e_3}{\|\dot{v} - g e_3\|}, \; \boldsymbol{b}_2 = \frac{\boldsymbol{b}_3 \times \boldsymbol{d}_1}{\|\boldsymbol{b}_3 \times \boldsymbol{d}_1\|}, \; \boldsymbol{b}_1 = \boldsymbol{b}_2 \times \boldsymbol{b}_3$$

provided $\boldsymbol{b}_3 \times \boldsymbol{d}_1 \neq 0$. This defines the rotation matrix ${}^{A}R_B$ as a function of $\dot{v}$ (the second derivative of ξ) and ψ. In this way, we write the angular velocity and the four inputs as functions of the position, velocity ($v = \dot{\xi}$), acceleration ($a = \ddot{\xi}$), jerk ($\gamma = \xi^{(iii)}$), and snap or the derivative of jerk ($\sigma = \xi^{(iv)}$). From these equations, it is possible to verify that there is a diffeomorphism between the 18×1 vector

$$\boldsymbol{X} = \left(\xi^{\mathrm{T}}, v^{\mathrm{T}}, a^{\mathrm{T}}, \gamma^{\mathrm{T}}, \sigma^{\mathrm{T}}, \psi^{\mathrm{T}}, \dot{\psi}^{\mathrm{T}}, \ddot{\psi}^{\mathrm{T}}\right)^{\mathrm{T}}$$

and the state augmented with the inputs and their derivatives

$$\left(\xi^{\mathrm{T}}, \dot{\xi}^{\mathrm{T}}, R, \Omega^{\mathrm{T}}, u_1, \dot{u}_1, \ddot{u}_1, \mathbf{u}_2^{\mathrm{T}}\right)^{\mathrm{T}} .$$

This property of differential flatness makes it easy to design trajectories that respect the dynamics of the underactuated system. Any four-times-differentiable trajectory in the space of flat outputs, $(\xi^{\mathrm{T}}(t), \psi(t))^{\mathrm{T}}$, corresponds to a feasible trajectory, one that satisfies the equations of motion. All inequality constraints of states and inputs can be expressed as functions of the flat outputs and their derivatives. This mapping to the space of flat outputs can be used to generate trajectories that minimize a cost functional formed by a weighted combination of the different flat outputs and

Avidbots Ex1016
Page 1349 of 2154

their derivatives

$$(\xi^p(t), \psi^p(t)) = \arg \min_{\xi(t), \psi(t)} \int_0^T L(\xi, v, a, \gamma, \sigma, \psi, \dot{\psi}, \ddot{\psi})\, \mathrm{d}t\,, \quad g(\xi(t), \psi(t)) \leq 0\,. \tag{52.48}$$

In [52.7], minimum snap trajectories were generated by minimizing a cost functional derived from the snap and the angular yaw acceleration with

$$L(\xi, \dot{\xi}, \ddot{\xi}, \dddot{\xi}, \ddddot{\xi}\psi, \dot{\psi}, \ddot{\psi}) = (1-\alpha)(\dddot{\xi})^4 + \alpha(\ddot{\psi})^2\,.$$

By suitable parameterizing trajectories with basis functions in the flat space and by considering linear inequalities in the flat space to model constraints on states and inputs (e.g., $u_1 \geq 0$), it is possible to turn this optimization into a quadratic program which can be solved in real-time for planning.

Finally, as shown in [52.45], it is possible to combine this controller with attitude only controllers to fly through vertical windows or land on inclined perches with close to zero normal velocity. A trajectory controller is used by the robot to build up momentum while the attitude controller enables reorientation when coasting with the generated momentum.

52.4.2 Trajectory Planning for Fixed-Wing Aircraft

For fixed wing vehicles, the critical constraints that influence path planning is the minimum turn radius and the maximum flight path angle. As shown in [52.29, 31], a good approximation for the minimum turn radius is given by

$$R_{\min} = \frac{|v_a|^2}{g} \tan\bar{\phi}\,,$$

where g is the acceleration of gravity at sea level, and $\bar{\phi}$ is the maximum allowable bank angle. From (52.27c), we see that a good approximation for the maximum flight path angle is

$$\bar{\gamma} = \sin^{-1}\left(\frac{\dot{h}_{\max}}{|v_a|}\right),$$

where $\dot{h}_{\max}$ is the maximum climb/descent rate of the airframe. Fixed wing aircraft are typically designed to fly at a specific airspeed where the lift-to-drag ratio is maximized, thereby minimizing fuel expenditure. Therefore path planners for fixed wing vehicles commonly assume a constant airspeed. In this section, we will assume zero wind conditions where airspeed equals ground speed. Therefore, the path planning problem for fixed wing vehicles is naturally posed as planning a constant velocity path between two configurations that satisfy the minimum turn radius and maximum flight path angle constraints.

If the maneuver is executed at a constant altitude, then a traditional Dubins car planner can be used to find a constant velocity path satisfying the turn rate constraints [52.46]. Using the Dubins car model for UAV path planning is a common practice in the literature [52.31, 47–50]. However, the constant altitude limitation can be removed by extending the traditional Dubins car model to the Dubins airplane model given by

$$\begin{aligned}\dot{n} &= V\cos\chi\,,\\ \dot{w} &= V\sin\chi\,,\\ \dot{\chi} &= u_1\,,\\ \dot{h} &= -V\sin u_2\,,\end{aligned}$$

where (n, w) is the inertial North–West position, V is the speed of the Dubins airplane, and in the case of zero ambient wind is $V = |v_a|$, and where $|u_1| \leq V/R_{\min}$, and $|u_2| \leq \bar{\gamma}$ [52.51].

The configuration of a Dubins airplane is given by $C = (n, w, h, \chi)$, where n and w are North–West inertial coordinates, h is the altitude, and χ is the course angle. The Dubins airplane planning problem is to find a minimum distance path between a start configuration C_s and an end configuration C_e that satisfies the constraints $|u_1| \leq V/R_{\min}$, and $|u_2| \leq \bar{\gamma}$.

Recall that Dubins car paths are composed of three segments: a turn segment that follows a circle with the minimum turn radius $R_{\min}$, a straight line segment, and finally another turn segment that again follows a circle with minimum turn radius [52.46]. The planning problem for the Dubins airplane is closely related to the planning problem for the Dubins car, with some variation due to the altitude component. As explained in [52.51, 52], Dubins airplane paths can be separated into three categories based on the altitude gain/loss between the start and end configurations, i. e., $\Delta h = |h_e - h_s|$.

If Δh is small enough, then the Dubins airplane can follow the standard Dubins car path while gaining or losing altitude at a constant rate that satisfies the flight path angle constraint and that allows the altitude difference to be realized. Such paths are termed *low-altitude paths* in [52.51]. If on the other hand, Δh is so large that flying the Dubins car path at the flight path angle constraints does not result in sufficient altitude gain/loss,

then the Dubins car path must be suitably modified. One alternative is to fly at the flight path angle constraint while making multiple complete orbits on the minimum turn radius helix at the beginning of the Dubins car path before proceeding along the straight line segment of the path. Of course, a similar strategy would be to make multiple complete orbits on the minimum turn radius helix at the end of the Dubins car path. Such paths are termed *high-altitude paths* in [52.51]. The middle case is when the flight path angle constraint does not allow sufficient altitude gain/loss following the Dubins car path, but where one complete orbit along the start or end circle of the Dubins car path at the maximum flight path angle results in more altitude gain/loss than is needed. In this case, a deviation can be placed in the Dubins car path to extend the path length to just the right amount so that the altitude gain/loss can be achieved while flying at the flight path angle constraint. These paths are termed *medium-altitude paths* in [52.51].

Following the notation defined in [52.52] let $L_{\text{car}}(C_\text{s}, C_\text{e}, R)$ denote the path length of the Dubins car path between the projection of the start configuration C_s onto the North–West plane, and the projection of the end configuration C_e onto the North–West plane, using R as the turn radius of the vehicle.

Low-Altitude Dubins Paths

The altitude gain between the start and end configuration is said to be *low altitude* if

$$|h_\text{e} - h_\text{s}| \leq L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) \tan\bar{\gamma} \,,$$

where the term on the right is the maximum altitude gain that can be obtained by flying at flight-path angle $\pm\bar{\gamma}$ for a distance of $L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min})$.

In the low-altitude case, the altitude gain between the start and end configurations can be achieved by flying the Dubins car path with a flight-path angle satisfying $|u_2| \leq \bar{\gamma}$. Therefore, the optimal flight-path angle can be computed by

$$u_2^* = \tan^{-1}\left(\frac{|h_\text{e} - h_\text{s}|}{L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min})}\right).$$

Medium-Altitude Dubins Paths

The altitude gain between the start and the end configuration is said to be *medium altitude* if

$$\begin{aligned} &L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) \tan\bar{\gamma} < |h_\text{e} - h_\text{s}| \\ &\quad \leq [L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi R_{\min}] \tan\bar{\gamma} \,, \end{aligned} \tag{52.49}$$

where the addition of the term $2\pi R_{\min}$ accounts for adding one orbit at radius $R_{\min}$ to the path length.

In the medium-altitude case, the altitude difference between the start and end configurations is too large to obtain by flying the Dubins car path at the flight-path angle constraint, but small enough that adding a full turn on the helix at the beginning or end of the path and flying so that $\gamma = \pm\bar{\gamma}$ results in more altitude gain/loss than is needed. As shown in [52.51], the minimum distance path is achieved by setting $\gamma = \text{sign}\,(h_\text{e} - h_\text{s})\,\bar{\gamma}$ and inserting an extra maneuver in the Dubins car path that extends the path length so that the altitude gain when $\gamma = \pm\bar{\gamma}$ is exactly $|h_\text{e} - h_\text{s}|$. While there are numerous possible ways to extend the path length, the method proposed in [52.52] is to add an additional intermediate arc to the start or end of the path, as shown in Figs. 52.8 and 52.9. If the start altitude is lower than the end altitude, then the intermediate arc is inserted immediately after the start helix. If on the other hand, the start altitude is higher than the end altitude, then the intermediate arc is inserted immediately before the end helix.

High-Altitude Dubins Paths

The altitude gain between the start and end configurations is said to be *high altitude* if

$$|h_\text{e} - h_\text{s}| > [L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi R_{\min}] \tan\bar{\gamma} \,.$$

In the high-altitude case, the altitude gain cannot be achieved by flying the Dubins car path within the flight-path angle constraints. As shown in [52.51], the minimum distance path is achieved when the flight-path angle is set at its limit of $\pm\bar{\gamma}$, and the Dubins car path is extended to facilitate the altitude gain. While there are many different ways to extend the Dubins car path, [52.52] suggests extending the path by spiraling a certain number of turns at the beginning or end of the path, and then by increasing the turn radius by the appropriate amount.

For UAV scenarios, the most judicious strategy is typically to spend most of the trajectory at as high an altitude as possible. Therefore, if the altitude at the end configuration is higher than the altitude at the start configuration, then the path will be extended by a climbing helix at the beginning of the path. If on the other hand, the altitude at the start configuration is higher than the end configuration, then the path will be extended by a descending helix at the end of the path. If multiple turns around the helix are required, then the turns could be split between the start and end helices and still result in the same path length. For high-altitude Dubins paths, the required number of turns in the helix will be the smallest integer k such that

$$\begin{aligned} &[L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi k R_{\min}] \tan\bar{\gamma} \leq |h_\text{e} - h_\text{s}| \\ &\quad < [L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi(k+1) R_{\min}] \tan\bar{\gamma} \,, \end{aligned} \tag{52.50}$$

Avidbots Ex1016
Page 1351 of 2154

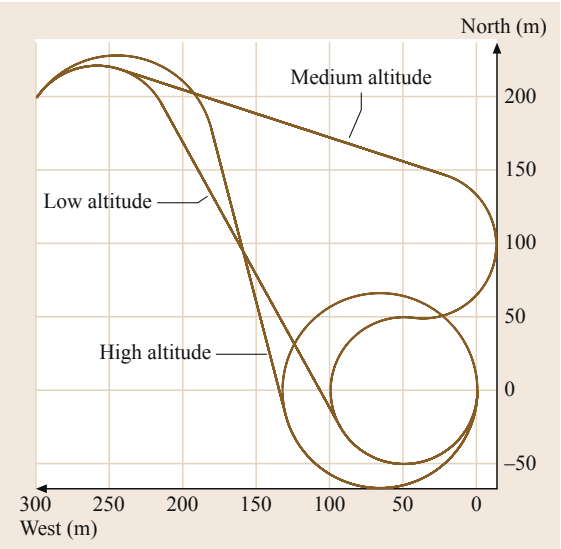


Fig. 52.8 A top down view of low altitude, medium altitude, and high-altitude Durbin airplane paths. The initial and end configurations are identical except for the final altitude

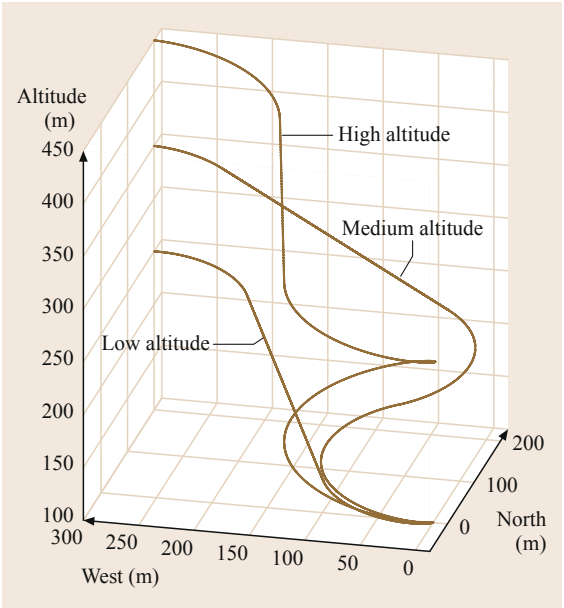


Fig. 52.9 A three-dimensional view of low altitude, medium altitude, and high-altitude Durbin airplane paths. The initial and end configurations are identical except for the final altitude

or in other words

$$k = \left\lfloor \frac{1}{2\pi R_{\min}} \left[\frac{|h_e - h_s|}{\tan \bar{\gamma}} - L_{car}(C_s, C_e, R_{\min}) \right] \right\rfloor ,$$

where $\lfloor x \rfloor$ is the floor function that rounds x down to the nearest integer. The radius of the start and end helices is then increased to $R^\star$ so that

$$\left[L_{car}(C_s, C_e, R^\star) + 2\pi k R^\star \right] \tan \bar{\gamma} = |h_e - h_s| \ . \tag{52.51}$$

Figures 52.8 and 52.9 show two perspective of three different Dubins airplane paths that start and end in the same configuration with the exception of the final altitude, which changes from 250 m for the low-altitude path, to 350 m for the medium-altitude path, to 450 m for the high-altitude path. A top down view is shown in Fig. 52.8. Note that for both the low-altitude and medium-altitude paths, that the minimum turn radius constraint is active, whereas for the high-altitude path, the turn radius in the turns is larger than the minimum turn radius constraint. A three-dimensional view is shown in Fig. 52.9. Note that for both the medium- and high-altitude paths, the flight path angle constraint is active, but that it is not active for low-altitude paths.

There are a variety of techniques that allow the control architecture described in Sect. 52.3.2 to follow Dubins airplane paths. A simple method is to parameterize the path using a path parameter σ. Suppose that the resulting parameterized path is given by $p(\sigma)$. At each sample time, the path parameter is advanced along the path so as to minimize the distance to the path from the aircraft

$$\sigma_{t+1} = \arg \min_{s \geq \sigma_t} \| \xi(t) - p(s) \| \ ,$$

where $\xi(t)$ is the inertial position of the aircraft. The commanded airspeed $|v_a^\star|$, course angle $\chi^\star$, and altitude $h^\star$ at time t are then given by the parameterized Dubins airplane path $p(\sigma_{t+1})$. This technique is similar to the guidance strategy suggested in [52.53]. Alternative method based on vector field methods are described in [52.54–56].

Avidbots Ex1016
Page 1352 of 2154

52.5 Estimating the Vehicle State

A critical aspect in implementation of real world aerial robotic vehicles is providing good estimates of the state of the vehicle. The key state estimates required for control of an aerial vehicle are associated with the rigid-body dynamics of its airframe, height, attitude, angular velocity, and linear velocity. Moreover, fixed-wing vehicle dynamics depend on the aerodynamics states of angle-of-attack and side-slip angle. Of these states, the attitude and angular velocity are the most important as they are the primary variable used in attitude control and flight-regulation of the vehicle. Angle-of-attack and side-slip angles are rarely estimated explicitly and are dealt with as internal dynamics in the control design rather than explicit outputs or disturbances.

The ubiquitous instrumentation carried by any aerial vehicle is an inertial measurement unit (IMU) often augmented by some form of height measurement, either acoustic, infra red, barometric, or laser based. Vehicles that fly in outside environments carry a GPS system, most of which now provide velocity estimation as well as position. Many fixed wing vehicles also include a pitot tube (for dynamic pressure) or an anemometer to measure forward velocity. Indoor aerial robotic vehicles in research laboratories are often flown in flight environments equipped with motion capture systems like VICON or Optitrack [52.5, 6]. Finally, many aerial robotic systems have also been equipped with exteroceptive sensor systems including Kinect 3D-range cameras, scanning laser rangefinder, integrated stereo camera systems, or simple monocular cameras. There is an active research field in developing 3-D-SLAM (simultaneous localization and mapping; Chap. 46.4) algorithms for mapping and localization of aerial robotic vehicles.

The rigid-body dynamic state is crucial to the control performance of an aerial robotic system and we will concentrate on this estimation problem in this chapter. A natural approach to state-estimation is to apply classical filter design to a coordinate representation of the rigid-body state. A good development of this approach for small-scale aerial robotic systems is presented in [52.31]. An alternative approach is to exploit the nonlinear structure of the rigid-body kinematics to develop nonlinear observers. This approach has been shown to generate simple, robust, and highly effective filters for small-scale aerial robotic systems, especially for the key problem of attitude estimation.

52.5.1 Estimating Attitude

A typical IMU includes a three-axis rate gyro, three-axis accelerometer, and three-axis magnetometer. The rate gyro measures angular velocity of $\{B\}$ relative to $\{A\}$ expressed in the body-fixed-frame of reference $\{B\}$

$$\Omega_{\text{IMU}} = \Omega + b_{\Omega} + \eta \in \{B\} ,$$

where η denotes additive measurement noise and b_{Ω} denotes a constant (or slowly time-varying) gyro bias. Generally, the gyroscopes installed on quadrotor vehicles are lightweight MEMS devices that are reasonably robust to noise and quite reliable.

The accelerometers (in a *strap down IMU configuration*) measure the instantaneous linear acceleration of $\{B\}$ due to exogenous force

$$a_{\text{IMU}} = R^{\text{T}}(\dot{v} + ge_3) + b_{\text{a}} + \eta_{\text{a}} \in \{B\} , \qquad (52.52)$$

where b_{a} is a bias term, η_{a} denotes additive measurement noise, and $\dot{v}$ is in the inertial frame. Accelerometers are highly susceptible to vibration and mounted on a typical aerial robotic platform they require significant low-pass mechanical and/or electrical filtering to be reliable. Most vehicles avionics will incorporate an analogue anti-aliasing filter on a MEMS accelerometer before the signal is sampled.

The magnetometers provide measurements of the ambient magnetic field

$$m_{\text{IMU}} = R^{\text{T}}\,{}^{A}m + B_m + \eta_b \in \{B\} ,$$

where ${}^{A}m$ is the Earth's magnetic field vector (expressed in the inertial frame), B_m is a body-fixed-frame expression for the local magnetic disturbance and η_b denotes measurement noise. The noise η_b is usually quite low for magnetometer readings, however, the local magnetic disturbance B_m can be very significant, especially if the sensor is placed near power cables for the motors.

In outdoor environments, the GPS is typically used to provide position, inertial velocity, and course angle. The GPS system consists of a constellation of 24 satellites that continuously orbit the earth at an altitude of 20 180 km. The position of a GPS receiver is determined by observing the time-of-flight of signals sent from the satellites and detected by the receiver. If accurate timing information is available at the receiver, then a minimum of three satellites signals are necessary to resolve the position of the receiver. However, for low-cost GPS receivers, timing information must also be received and this requires a minimum of four satellite signals. A variety of factors effect the accuracy of GPS estimates. The dominant sources of error include inaccurate satellite orbital data, inaccuracy in the satellite clocks, variable signal delay as it passes through

Avidbots Ex1016
Page 1353 of 2154

the ionosphere, weather conditions near the earth surface, and multipath reflections from nearby buildings and mountains. Total GPS bias error is approximately 5–10 m. Modern GPS receivers exploit doppler shift in the carrier phase of the received signals to estimate the course angle χ and the ground speed.

The accelerometers and magnetometers can be used to provide absolute attitude information on the vehicle while the rate gyroscope provides complementary angular velocity measurements. The attitude information in the magnetometer signal is straightforward to understand; in the absence of noise and bias then m_{IMU} provides a body-fixed frame measurement of ${}^{A}R^{T}m$, and consequently constrains two degrees of freedom in the rotation R. The accelerometer can also be used as long as the component of acceleration associated with the inertial motion of the vehicle is compensated. The simplest way in which to do this is to use an absolute external signal such as GPS to estimate the vehicle acceleration

$$a_{\text{CTD}} = a_{\text{IMU}} - \hat{R}^{\text{T}}\ddot{\xi}_{\text{GPS}} \,,$$

where a_{CTD} is the corrected acceleration. In this case, $a_{\text{CTD}} \approx R^{\text{T}}e_3$ and this provides attitude information. It is important to note that in the complementary filter proposed below (52.53), only the low-frequency component of a_{CTD} is required. As such, the potential noise in the second derivative of a GPS signal $\ddot{\xi}_{\text{GPS}}$ is not necessarily the problem that may be initially feared. The key to good filter performance is to ensure that the cross over frequency of the complimentary sensitivity is low enough that any phase distortion associated with low-pass filtering of $\ddot{\xi}_{\text{GPS}}$ does not effect the estimate.

Remark 52.1
There may be issues in applying the correction $\hat{R}^{\text{T}}\ddot{\xi}_{\text{GPS}}$ if there is a delay in receiving the GPS signals. This can be addressed, at the cost of storing IMU data, by combining a short time prediction along with a time-lagged observer [52.57].

In the case where GPS or an external measure of the vehicles motion is not available, there are a number of alternative techniques available to use the accelerometer information for attitude estimation. For fixed-wing vehicles, the most important disturbance is due to centripetal acceleration due to the vehicle turning. It is possible to model the centripetal acceleration using airspeed and angular velocity and compensate using a feedforward term [52.58, 59]. In the case of rotorcraft in mostly hover flight conditions, it turns out that using the raw IMU accelerometer readings still works extremely well [52.60–62] due to the disturbance caused by the aerodynamic drag term $\Delta = -Dv$ (52.10) [52.21, 24, 25, 63]. In this case the corrected accelerometer a_{CTD} used in the filter (52.53) below is just the raw IMU accelerometer measurement.

The attitude kinematics of the quadrotor are given by (52.2c). Let $\hat{R}$ denote an estimate for attitude R of the quadrotor vehicle. The following observer [52.61, 62] fuses accelerometer, magnetometer, and gyroscope data as well as other direct attitude estimates R_{E} (such as provided by a VICON or other external measurement system) should they be available

$$\dot{\hat{R}} := \hat{R}\left(\Omega_{\text{IMU}} - \hat{b}\right)_{\times} - \Upsilon \,, \tag{52.53a}$$

$$\dot{\hat{b}} := k_b \Upsilon \,, \tag{52.53b}$$

$$\begin{aligned}\Upsilon := &\left(\frac{k_a}{g^2}\left[(\hat{R}^{\text{T}}e_3)\times a_{\text{CTD}}\right]\right.\\ &\left.+\frac{k_m}{{}^{A}m^2}\left[(\hat{R}^{\text{T}A}m)\times m_{\text{IMU}}\right]\right)_{\times}\\ &+k_E\mathbb{P}_{\mathfrak{so}(3)}\left(\hat{R}R_{\text{E}}^{\text{T}}\right),\end{aligned} \tag{52.53c}$$

where k_a, k_m, k_E, and k_b are arbitrary nonnegative observer gains and $\mathbb{P}_{\mathfrak{so}(3)}(M) = (M - M^{\text{T}})/2$ is the Euclidean matrix projection onto the skew-symmetric matrices. If any one of the measurements in the innovation Υ are not available or unreliable, then the corresponding gain should be set to zero in the observer. Note that both the attitude $\hat{R}$ and the bias corrected angular velocity $\hat{\Omega} = \Omega_{\text{IMU}} - \hat{b}$ are estimated by this observer. The observer (52.53) has been extensively studied in the literature [52.61, 62] and shown to converge exponentially (both theoretically and experimentally) to the desired attitude estimate of attitude with $\hat{b}$ converging to the gyroscope bias b. The filter has a complementary nature, using the high-frequency part of the gyroscope signal and the low-frequency parts of the magnetometer, accelerometer, and external attitude measurements [52.61]. The roll off frequencies associated with each of these signals is given by the gains k_a, k_m, and k_E in rad/s.

52.5.2 Estimating Velocity and Position

Estimating velocity and position is a straightforward process if the vehicle is equipped with a GPS system. In this case, the dynamics (52.2a) and (52.2b) are linear, and a linear filter can be used. Let $\hat{\xi}$ denote the estimate of position and $\hat{v}$ denote the estimate of velocity. A sim-

ple linear filter is given by

$$\dot{\hat{\xi}} = \hat{v} - k_x(\hat{\xi} - \xi_{\mathrm{GPS}}) ,$$
$$\dot{\hat{v}} = \hat{R}^{\mathrm{T}} a_{\mathrm{IMU}} - g e_3 - k_v(\hat{\xi} - \xi_{\mathrm{GPS}}) ,$$

for gains $k_x, k_v > 0$. As long as the attitude estimate $\hat{R}$ is accurate, this filter design is highly robust. It is easily possible to add bias estimates for the accelerometer if desired. Equally well it is straightforward to use Kalman filter techniques to tune the gains k_x and k_v in real-time if that is considered advisable. In practice, for small aerial robotic systems, the noise characteristics of the measurements are so poor that it is often best to use a constant gain filter rather than introduce the additional complexity and potential instability of the Riccati equation associated with a Kalman filter.

In the absence of GPS, estimating position depends on the availability of additional sensor systems. Since most aerial robots are equipped with a barometer to provide an estimate of height and this can be utilized for an estimate of the vertical motion of the vehicle

$$\dot{\hat{h}} = \hat{v}_z - k_h(\hat{h} - h) , \tag{52.54a}$$
$$\dot{\hat{v}}_z = e_3^{\mathrm{T}} \hat{R}^{\mathrm{T}} a_{\mathrm{IMU}} - g - k_{v_z}(\hat{h} - h) , \tag{52.54b}$$

where $k_h, k_{v_z} > 0$ are positive gains.

For quadrotor vehicles, there is also the possibility of exploiting the linear drag aerodynamic forces (52.10) associated with blade flapping and induced drag to estimate horizontal velocity for a vehicle in near hover conditions [52.24]. Define a projector matrix

$$\mathbb{P}_h := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{52.55}$$

that takes the first two components of a vector. Then the horizontal component of the *inertial* acceleration can be measured by

$${}^A a_h := \mathbb{P}_h \, {}^A a = \mathbb{P}_h R a \approx \mathbb{P}_h \hat{R} a , \tag{52.56}$$

where we assume that the estimate $\hat{R}$ is close to R. If we assume the vehicle is only slowly varying in height one has $v_z \approx 0$ in comparison to horizontal velocity

$$v_h \approx \mathbb{P}_h^{\mathrm{T}} v .$$

Furthermore, the thrust $T \approx mg$ must compensate the weight of the vehicle. Recalling (52.10) and taking just the horizontal component one has

$${}^A a_h \approx -g \mathbb{P}_h \hat{R} e_3 - g \mathbb{P}_h \hat{R} D R^{\mathrm{T}} \mathbb{P}_h^{\mathrm{T}} v_h . \tag{52.57}$$

If the attitude filter estimate is good and the assumptions about vehicle motion hold then (52.56) and (52.57) can be solved for an estimate of v_h

$$v_h \approx -\frac{1}{g} \left(\mathbb{P}_h \hat{R} D \hat{R}^{\mathrm{T}} \mathbb{P}_h^{\mathrm{T}} \right)^{-1} \left({}^A a_h + g \mathbb{P}_h \hat{R} e_3 \right) . \tag{52.58}$$

Equation (52.58) provides a measurement of the horizontal velocity, however, since it directly incorporates the unfiltered accelerometer readings it is generally too noisy to be of much use. Its low-frequency content can, however, be used to drive a velocity complementary observer that uses the attitude estimate and the system model (52.2b). Let $\hat{v}_h$ be an estimate of the horizontal component of the inertial velocity of the vehicle then an observer is given by

$$\dot{\hat{v}}_h = -g \mathbb{P}_h^{\mathrm{T}} \left(\hat{R} e_3 + \hat{R} D \hat{R}^{\mathrm{T}} \mathbb{P}_h^{\mathrm{T}} \hat{v}_h \right) - k_{\mathrm{w}} (\hat{v}_h - v_h) , \tag{52.59}$$

where v_h is given by (52.58). The gain $k_{\mathrm{w}} > 0$ provides a tuning parameters that indicates the roll-off frequency for the information from $\hat{v}_h$ that is used in the filter. It also uses estimated velocity $\hat{v}_h$ to provide an approximation of the more correct $RDR^{\mathrm{T}} \mathbb{P}_h^{\mathrm{T}} v_h$ term in the feedforward velocity estimate, however, since the underlying dynamics associated with this term is stable, then the observer is stable even with this approximation. It is found that this observer works remarkably well in practice for quadrotor vehicles that are not engaged in acrobatic maneuvers [52.21, 24, 25, 63].

52.6 Conclusion

This chapter has focused on the fundamental technology associated with making an aerial robotic system operate effectively and efficiently. In particular, we have focused on the design of control and navigation algorithms, and along with that the associated questions of modeling and state estimation. Once a robust and reliable aerial robotic platform is available, the range of potential applications for unmanned aerial systems is vast. An indication of the potential is given by a list of application categories that are already under consideration [52.64, 65]: *Remote sensing* such as pipeline spotting, power-line monitoring, volcanic sampling, mapping, meteorology, geology, and agriculture [52.66, 67], as well as unexploded mine detection [52.68].

Disaster response such as chemical sensing, flood monitoring, and wildfire management. *Surveillance* such as law enforcement, traffic monitoring, coastal and maritime patrol, and border patrols [52.69]. *Search and rescue* in low-density or hard-to-reach areas. *Transportation* including small and large cargo transport, and possibly passenger transport. *Communications* as permanent or *ad hoc* communication relays for voice and data transmission, as well as broadcast units for television or radio. *Payload delivery* for a wide range of applications including agriculture, firefighting, and even logistics of product delivery. *Image acquisition* for cinematography and real-time entertainment. It is no surprise that aerial robotics is one of the most dynamic and exciting fields of robotics research at the moment.

Video-References

VIDEO 436 Autopilot using total energy control
available from http://handbookofrobotics.org/view-chapter/52/videodetails/436

VIDEO 437 Dubins airplane
available from http://handbookofrobotics.org/view-chapter/52/videodetails/437

References

52.1 R.C. Michelson: International aerial robotics competition – The world's smallest intelligent flying machines, 13th RPVs/UAVs International Conference (1998), pp. 31.1–31.9

52.2 L.R. Newcome: *Unmanned Aviation, a Brief History of Unmanned Aerial Vehicles* (American Institute of Aeronautics and Astronautics, Reston 2004)

52.3 M.A. Garratt, J.S. Chahl: Vision-based terrain following for an unmanned rotorcraft, J. Field Robotics **25**(4/5), 284–301 (2008)

52.4 M. Garratt, H. Pota, A. Lambert, S. Eckersley-Maslin, C. Farabet: Visual tracking and lidar relative positioning for automated launch and recovery of an unmanned rotorcraft from ships at sea, Naval Eng. J. **121**(2), 99–110 (2009)

52.5 Vicon Motion Systems Ltd: http://www.vicon.com/

52.6 NaturalPoint Inc.: OptiTrack, http://www.naturalpoint.com/optitrack/

52.7 D. Mellinger, V. Kumar: Minimum snap trajectory generation and control for quadrotors, Proc. Int. Conf. Robotics Autom. (ICRA) (2011)

52.8 M. Hehn, R. D'Andrea: Quadrocopter trajectory generation and control, IFAC World Congress (2011) pp. 1485–1491

52.9 M.W. Mueller, R. D'Andrea: A model predictive controller for quadrocopter state interception, European Control Conference (2013) pp. 1383–1389

52.10 M.W. Mueller, M. Hehn, R. D'Andrea: A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013) pp. 3480–3486

52.11 J. Thomas, J. Polin, K. Sreenath, V. Kumar: Avian-inspired grasping for quadrotor micro uavs, ASME Int. Des. Eng. Tech. Conf. (IDETC) (2013)

52.12 M. Turpin, N. Michael, V. Kumar: Trajectory design and control for aggressive formation flight with quadrotors, Auton. Robots **33**(1/2), 143–156 (2012)

52.13 S. Shen, Y. Mulgaonkar, N. Michael, V. Kumar: Vision-based state estimation for autonomous rotorcraft MAVs in complex environments, Proc. IEEE Int. Conf. Robotics Autom. (2013)

52.14 S. Shen, Y. Mulgaonkar, N. Michael, V. Kumar: Initialization-free monocular visual-inertial estimation with application to autonomous mavs, Int. Symp. Exp. Robotics (ISER) (2014)

52.15 T. Hamel, R. Mahony, R. Lozano, J. Ostrowski: Dynamic modelling and configuration stabilization for an X4-flyer, Proc. Int. Fed. Autom. Control Symp. (IFAC) (2002)

52.16 S. Bouabdallah, P. Murrieri, R. Siegwart: Design and control of an indoor micro quadrotor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 5 (2004) pp. 4393–4398

52.17 P.-J. Bristeau, P. Martin, E. Salaün, N. Petit: The role of propeller aerodynamics in the model of a quadrotor uav, Proc. Eur. Control Conf. (2009) pp. 683–688

52.18 H. Huang, G.M. Hoffmann, S.L. Waslander, C.J. Tomlin: Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering, IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 3277–3282

52.19 P. Pounds, R. Mahony, P. Corke: Modelling and control of a large quadrotor robot. Control Eng, Pract. **18**(7), 691–699 (2010)

52.20 M. Bangura, R. Mahony: Nonlinear dynamic modeling for high performance control of a quadrotor, Australas. Conf. Robotics Autom. (2012)

52.21 R. Mahony, V. Kumar, P. Corke: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor, Robotics Autom. Mag. **19**(3), 20–32 (2012)

52.22 R.W. Prouty: *Helicopter Performance, Stability and Control* (Krieger, Malabar 1995), reprint with additions

52.23 J.G. Leishman: *Principles of helicopter aerodynamics*, Cambridge Aerospace Series (Cambridge University Press, Cambridge 2000)
52.24 P. Martin, E. Salaun: The true role of accelerometer feedback in quadrotor control, Proc. IEEE Int. Conf. Robotics Autom. (2010) pp. 1623–1629
52.25 D. Abeywardena, S. Kodagoda, G. Dissanayake, R. Munasinghe: Improved state estimation in quadrotor mavs: A novel drift-free velocity estimator, IEEE Robotics Autom. Mag. **20**(4), 32–39 (2013)
52.26 R. Leishman, J. Macdonald, R.W. Beard, T.W. McLain: Quadrotors and accelerometers: State estimation with an improved dynamic model. IEEE Control Syst, Mag. **34**(1), 28–41 (2014)
52.27 A. Bramwell, G. Done, D. Balmford: *Bramwell's Helicopter Dynamics* (Butterworth Heinenmann, Woburn 2001)
52.28 J.-L. Boiffier: *The Flight Dynamics: The Equations* (Wiley, Chichester 1998)
52.29 B.L. Stevens, F.L. Lewis: *Aircraft Control and Simulation*, 2nd edn. (Wiley, Hoboken 2003)
52.30 L.H. Carter, J.S. Shamma: Gain-scheduled bank-to-turn autopilot design using linear parameter varying transformations. J. Guid, Control Dyn. **19**(5), 1056–1063 (1996)
52.31 R.W. Beard, T.W. McLain: *Navigation, Guidance, and Control of Small Unmanned Aircraft* (Princeton University Press, Princeton 2012)
52.32 M.-D. Hua, T. Hamel, P. Morin, C. Samson: A control approach for thrust-propelled underactuated vehicles and its application to vtol drones, IEEE Trans. Autom. Control **54**(8), 1837–1853 (2009)
52.33 M. Bangura, H. Lim, H. Jin Kim, R. Mahony: Aerodynamic power control for multirotor aerial vehicles, Proc. IEEE Int. Conf. Robotics Autom. (2014), Paper MoB03.2.
52.34 P. Pounds, R. Mahony, P. Corke: Design of a static thruster for micro air vehicle rotorcraft, ASCE J. Aerosp. Eng. **22**(1), 85–94 (2009)
52.35 T. Lee, M. Leok, N.H. McClamroch: Geometric tracking control of a quadrotor UAV on $se(3)$, Proc. IEEE Conf. Decis. Control (2010)
52.36 M. Bangura, R. Mahony: Real-time model predictive control for quadrotors, Proc. IFAC World Conf. (2014)
52.37 H.K. Khalil: *Nonlinear Systems*, 2nd edn. (Prentice Hall, Upper Saddle River 1996)
52.38 D. Pucci: Flight dynamics and control in relation to stall, IEEE Am. Control Conf. (ACC) (2012) pp. 118–124
52.39 S. Devasia, D. Chen, B. Paden: Nonlinear inversion-based output tracking, IEEE Trans. Autom. Control **41**(7), 930–942 (1996)
52.40 J.-F. Magni, S. Bennani, J. Terlouw: *Robust Flight Control: A Design Challenge*, Lecture Notes in Control and Information Sciences, Vol. 224 (Springer, Berlin, Heidelberg 1997)
52.41 E. Slatyer, R. Mahony, P. Corke: Terrain following using wide field optic flow, Proc. Australas. Conf. Robotics Autom. (ACRA) (2010)
52.42 A.A. Lambregts: Vertical fight path and speed control autopilot design using total energy principles, Proc. AIAA Guid. Control Conf. (1983) pp. 559–569
52.43 L.F. Faleiro, A.A. Lambregts: Analysis and tuning of a total energy control system control law using eigenstructure assignment, Aerosp. Sci. Technol. **3**(3), 127–140 (1999)
52.44 M.J. Van Nieuwstadt, R.M. Murray: Real-time trajectory generation for differentially flat systems, Int. J. Robust Nonlinear Control **8**, 995–1020 (1998)
52.45 D. Mellinger, N. Michael, V. Kumar: Trajectory generation and control for precise aggressive maneuvers with quadrotors, Int. J. Robotics Res. **31**, 664–674 (2012)
52.46 L.E. Dubins: On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents, Am. J. Math. **79**, 497–516 (1957)
52.47 G. Yang, V. Kapila: Optimal path planning for unmanned air vehicles with kinematic and tactical constraints, Proc. IEEE Conf. Decis. Control (2002) pp. 1301–1306
52.48 A. Rahmani, X.C. Ding, M. Egerstedt: Optimal motion primitives for multi-UAV convoy protection, Proc. Int. Conf. Robotics Autom. (2010) pp. 4469–4474
52.49 S. Hosak, D. Ghose: Optimal geometrical path in 3D with curvature constraint, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 113–118
52.50 H. Yu, R.W. Beard: A vision-based collision avoidance technique for micro air vehicles using local-level frame mapping and path planning, Auton. Robots **34**(1/2), 93–109 (2013)
52.51 H. Chitsaz, S.M. LaValle: Time-optimal paths for a Dubins airplane, Proc. 46th IEEE Conf. Decis. Control (2007) pp. 2379–2384
52.52 M. Owen, R.W. Beard, T.W. McLain: Implementing dubins airplane paths on fixed-wing UAVs. In: *Handbook of Unmanned Aerial Vehicles*, ed. by G.J. Vachtsevanos, K.P. Valavanis (Springer, Berlin, Heidelberg 2014) pp. 1677–1702
52.53 S. Park, J. Deyst, J.P. How: Performance and lyapunov stability of a nonlinear path-following guidance method, AIAA J. Guid. Control Dyn. **30**(6), 1718–1728 (2007)
52.54 D.R. Nelson, D.B. Barber, T.W. McLain, R.W. Beard: Vector field path following for miniature air vehicles, IEEE Trans. Robotics **37**(3), 519–529 (2007)
52.55 D.A. Lawrence, E.W. Frew, W.J. Pisano: Lyapunov vector fields for autonomous unmanned aircraft flight control, AIAA J. Guid. Control Dyn. **31**(5), 1220–12229 (2008)
52.56 V.M. Goncalves, L.C.A. Pimenta, C.A. Maia, B.C.O. Durtra, G.A.S. Pereira: Vector fields for robot navigation along time-varying curves in n-dimensions, IEEE Trans. Robotics **26**(4), 647–659 (2010)
52.57 A. Khosravian, J. Trumpf, R. Mahony, T. Hamel: Velocity aided attitude estimation on $so(3)$ with sensor delay, Proc. Conf. Decis. Control (2014)
52.58 M. Euston, P. Coote, R. Mahony, J. Kim, T. Hamel: A complementary filter for attitude estimation of a fixed-wing UAV, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 340–345

Avidbots Ex1016
Page 1357 of 2154

52.59 R. Mahony, M. Euston, J. Kim, P. Coote, T. Hamel: A nonlinear observer for attitude estimation of a fixed-wing UAV without GPS measurements, Trans. Inst. Meas. Control **33**(6), 699–717 (2011)
52.60 R. Mahony, T. Hamel, J.-M. Pflimlin: Complementary filter design on the special orthogonal group *SO*(3), Proc. IEEE Conf. Decis. Control (CDC) (2005) pp. 1477–1484
52.61 R. Mahony, T. Hamel, J.-M. Pflimlin: Non-linear complementary filters on the special orthogonal group, IEEE Trans. Autom. Control **53**(5), 1203–1218 (2008)
52.62 S. Bonnabel, P. Martin, P. Rouchon: Non-linear symmetry-preserving observers on lie groups, IEEE Trans. Autom. Control **54**(7), 1709–1713 (2009)
52.63 J. Macdonald, R. Leishman, R. Beard, T. McLain: Analysis of an improved imu-based observer for multirotor helicopters, J. Intell Robotic Syst. **74**(3/4), 1049–1061 (2014)
52.64 D. Hughes: Uavs face hurdles in gaining access to civil airspace, Aviation Week (2007)
52.65 K.C. Wong, C. Bil, G. Gordon, P.W. Gibbens: *Study of the Unmanned Aerial Vehicle (UAV) Market in Australia*, Tech. Rep. (Aerospace Technology Forum Report, Sydney 1997)
52.66 R. Sugiura, N. Noguchi, K. Ishii, H. Terao: Development of remote sensing system using an unmanned helicopter, J. Jap. Soc. Agricult. Mach. **65**(1), 53–61 (2003)
52.67 R. Sugiura, T. Fukagawa, N. Noguchi, K. Ishii, Y. Shibata, K. Toriyama: Field information system using an agricultural helicopter towards precision farming, IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2003) pp. 1073–1078
52.68 K. Schutte, H. Sahli, D. Schrottmayer, F.J. Varas: Arc: A camcopter based mine field detection system, 5th Int. Airborne Remote Sens. Conf. (2001)
52.69 S.E. Wright: Uavs in community police work, AIAA Infotechs., Aerospace (2005)

Avidbots Ex1016
Page 1358 of 2154

Multimedia Contents

53. Multiple Mobile Robot Systems

Lynne E. Parker, Daniela Rus, Gaurav S. Sukhatme

Within the context of multiple *mobile, and networked* robot systems, this chapter explores the current state of the art. After a brief introduction, we first examine architectures for multirobot cooperation, exploring the alternative approaches that have been developed. Next, we explore communications issues and their impact on multirobot teams in Sect. 53.3, followed by a discussion of networked mobile robots in Sect. 53.4. Following this we discuss swarm robot systems in Sect. 53.5 and modular robot systems in Sect. 53.6. While swarm and modular systems typically assume large numbers of homogeneous robots, other types of multirobot systems include heterogeneous robots. We therefore next discuss heterogeneity in cooperative robot teams in Sect. 53.7. Once robot teams allow for individual heterogeneity, issues of task allocation become important; Sect. 53.8 therefore discusses common approaches to task allocation. Section 53.9 discusses the challenges of multirobot learning, and some representative approaches. We outline some of the typical application domains which serve as test beds for multirobot systems research in Sect. 53.10. Finally, we conclude in Sect. 53.11 with some summary remarks and suggestions for further reading.

53.1 **History** 1336

53.2 **Architectures for Multirobot Systems** ... 1337
- 53.2.1 The Nerd Herd 1337
- 53.2.2 The ALLIANCE Architecture 1338
- 53.2.3 The Distributed Robot Architecture 1339

53.3 **Communication** 1339

53.4 **Networked Mobile Robots** 1340
- 53.4.1 Overview 1341
- 53.4.2 State of the Art and Potential .. 1342
- 53.4.3 Research Challenges 1344
- 53.4.4 Control 1346
- 53.4.5 Communication for Control 1347
- 53.4.6 Communication for Perception . 1347
- 53.4.7 Control for Perception 1349
- 53.4.8 Control for Communication 1350

53.5 **Swarm Robots** 1351

53.6 **Modular Robotics** 1354
- 53.6.1 Chain Systems 1354
- 53.6.2 Lattice Systems 1354
- 53.6.3 Truss Systems 1356
- 53.6.4 Free-Form Systems 1356
- 53.6.5 Self-Assembling Systems 1356

53.7 **Heterogeneity** 1357

53.8 **Task Allocation** 1359
- 53.8.1 Taxonomy for Task Allocation ... 1359
- 53.8.2 Representative Approaches 1360

53.9 **Learning** 1361

53.10 **Applications** 1362
- 53.10.1 Foraging and Coverage 1362
- 53.10.2 Flocking and Formations 1362
- 53.10.3 Object Transportation and Cooperative Manipulation . 1363
- 53.10.4 Multitarget Observation 1364
- 53.10.5 Traffic Control and Multirobot Path Planning 1365
- 53.10.6 Soccer 1365

53.11 **Conclusions and Further Reading** 1366

Video-References 1366

References 1367

Avidbots Ex1016
Page 1359 of 2154

Researchers generally agree that multirobot systems have several advantages over single-robot systems [53.1, 2]. The most common motivations for developing multirobot system solutions are that:

1. The task complexity is too high for a single robot to accomplish.
2. The task is inherently distributed.
3. Building several resource-bounded robots is much easier than having a single powerful robot.
4. Multiple robots can solve problems faster using parallelism.
5. The introduction of multiple robots increases robustness through redundancy.

The issues that must be addressed in developing multirobot solutions are dependent upon the task requirements and the sensory and effector capabilities of the available robots.

The types of robots considered in the study of multiple *mobile* robot systems are those robots that move around in the environment, such as ground vehicles, aerial vehicles, or underwater vehicles. This chapter focuses specifically on the interaction of multiple *mobile* robots, as distinguished from other types of multirobot interaction. For example, a special case of multiple mobile robot systems are the *reconfigurable* or *modular* robots that interconnect with each other for the purposes of navigation or manipulation. Algorithmic aspects of these systems are covered in Sect. 53.5, while hardware aspects are covered in Chap. 22. *Networked* robotics is very closely related to multiple mobile robot systems; the focus in networked robotics is on systems of robots, sensors, embedded computers, and human users that are all connected by networked communication. Another variant of multirobot cooperation is multiple manipulator arm cooperation. Chapter 39 describes these systems in detail.

53.1 History

Since the earliest work on multiple mobile robot systems in the 1980s, the field has grown significantly, and covers a large body of research. At the most general level, approaches to multiple mobile robot systems fall into one of two broad categories: *collective swarm* systems and *intentionally cooperative* systems. *Collective swarm* systems are those in which robots execute their own tasks with only minimal need for knowledge about other robot team members. These systems are typified by the assumption of a large number of homogeneous mobile robots, in which robots make use of local control laws to generate globally coherent team behaviors, with little explicit communication among robots. On the other hand, robots in *intentionally cooperative* systems have knowledge of the presence of other robots in the environment and act together based on the state, actions, or capabilities of their teammates in order to accomplish the same goal. Intentionally cooperative systems vary in the extent to which robots take into account the actions or state of other robots, and can lead to either strongly or weakly cooperative solutions [53.3]. *Strongly cooperative* solutions require robots to act in concert to achieve the goal, executing tasks that are not trivially serializable. Typically, these approaches require some type of communication and synchronization among the robots. *Weakly cooperative* solutions allow robots to have periods of operational independence, subsequent to coordinating their selection of tasks or roles. Intentionally cooperative multirobot systems can deal with heterogeneity in the robot team members, in which team members vary in their sensor and effector capabilities. In these teams, the coordination of robots can be very different from collective swarm approaches, since robots are no longer interchangeable.

Most of the work specific to multiple mobile robot cooperation can be categorized into a set of key topics of study. These topics, which are the foci of this chapter, include *architectures*, *communication*, *swarm robots*, *heterogeneity*, *task allocation*, and *learning*. *Architectures* and *communication* in multirobot systems are relevant for all types of multirobot systems, as these approaches specify how the robot team members are organized and interact. *Swarm robots* is a particular type of multirobot system, typified by large numbers of homogeneous robots that interact implicitly with each other. Such systems are often contrasted with *heterogeneous* robots, in which team members may vary significantly in their capabilities. When robots vary in capabilities, challenges arise in determining which robots should perform which tasks – a challenge commonly referred to as *task allocation*. Finally *learning* in multirobot teams is of particular interest in designing teams that are adaptive over time and can learn new behaviors. Illustrating the advances in each of these areas often takes place in a set of representative application domains; these *applications* are the final major topic of discussion in this chapter.

53.2 Architectures for Multirobot Systems

The design of the overall control architecture for the multirobot team has a significant impact on the robustness and scalability of the system. Robot architectures for multirobot teams are composed of the same fundamental components as in single-robot systems, as described in Chap. 12. However, they also must address the interaction of robots and how the group behavior will be generated from the control architectures of the individual robots in the team. Several different philosophies for multirobot team architectures are possible; the most common are *centralized*, *hierarchical*, *decentralized*, and *hybrid*.

Centralized architectures that coordinate the entire team from a single point of control are theoretically possible [53.4], although often practically unrealistic due to their vulnerability to a single point of failure, and due to the difficulty of communicating the entire system state back to the central location at a frequency suitable for real-time control. Situations in which these approaches are relevant are cases in which the centralized controller has a clear vantage point from which to observe the robots, and can easily broadcast group messages for all robots to obey [53.5].

Hierarchical architectures are realistic for some applications. In this control approach, each robot oversees the actions of a relatively small group of other robots, each of which in turn oversees yet another group of robots, and so forth, down to the lowest robot, which simply executes its part of the task. This architecture scales much better than centralized approaches, and is reminiscent of military command and control. A point of weakness for the hierarchical control architecture is recovering from failures of robots high in the control tree.

Decentralized control architectures are the most common approach for multirobot teams, and typically require robots to take actions based only on knowledge local to their situation. This control approach can be highly robust to failure, since no robot is responsible for the control of any other robot. However, achieving global coherency in these systems can be difficult, because high-level goals have to be incorporated into the local control of each robot. If the goals change, it may be difficult to revise the behavior of individual robots.

Hybrid control architectures combine local control with higher-level control approaches to achieve both robustness and the ability to influence the entire team's actions through global goals, plans, or control. Many multirobot control approaches make use of hybrid architectures.

A plethora of multirobot control architectures have been developed over the years. We focus here on three early approaches that illustrate the spectrum of control architectures. The first, the Nerd Herd, is representative of a pure *swarm* robotics approach using large numbers of homogeneous robots. The second, ALLIANCE, is representative of a behavior-based approach that enables coordination and control of possibly heterogeneous robots without explicit coordination. The third, distributed robot architecture (DIRA), is a hybrid approach that enables both robot autonomy and explicit coordination in possibly heterogeneous robot teams.

53.2.1 The Nerd Herd

One of the first studies of social behaviors in multirobot teams was conducted by *Matarić* [53.6], with results being demonstrated on the Nerd Herd team of 20 identical robots (shown in Fig. 53.1). This work is an example of swarm robotic systems, as described further in Sect. 53.4. The decentralized control approach was based on the subsumption architecture (Chap. 12), and assumed that all robots were homogeneous, but with relatively simple individual capabilities, such as detecting obstacles and *kin* (i. e., other robot team members). A set of basic social behaviors were defined and demonstrated, including obstacle avoidance, homing, aggregation, dispersion, following, and safe wandering. These basic behaviors were combined in various ways to yield more composite social behaviors, including flocking (composed of safe wandering, aggregation, and dispersion), surrounding (composed of safe wandering, following, and aggregation), herding (composed of safe wandering, surrounding, and flocking), and foraging (composed of safe wandering, dispersion, following, homing, and flocking). The behaviors were

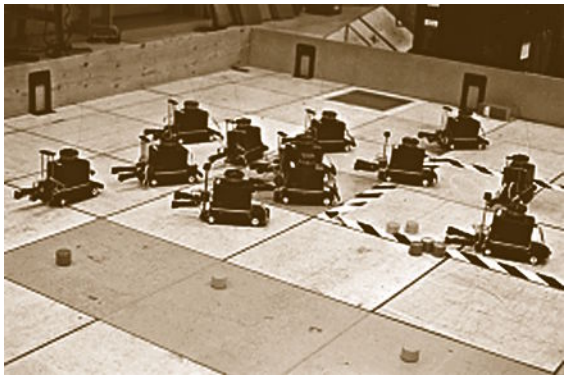


Fig. 53.1 The Nerd Herd robots

implemented as rules, such as the following rule for aggregate:

```
Aggregate:
  If agent is outside aggregation
  distance
    turn toward aggregation centroid
    and go.
  Else
    stop.
```

This work showed that collective behaviors could be generated through the combination of lower-level basic behaviors. Related work on this project studied issues such as using bucket brigades to reduce interference [53.7], and learning [53.8].

53.2.2 The ALLIANCE Architecture

Another early work in multirobot team architectures is the ALLIANCE architecture (Fig. 53.2), developed by *Parker* [53.9] for fault-tolerant task allocation in heterogeneous robot teams. This approach builds on the subsumption architecture by adding behavior sets and motivations for achieving action selection without explicit negotiations between robots. Behavior sets group low-level behaviors together for the execution of a particular task. The motivations consist of levels of impatience and acquiescence that can raise and lower a robot's interest in activating a behavior set corresponding to a task that must be accomplished.

In this approach, the initial motivation to perform a given behavior set is set to zero. Then, at each time step, the motivation level is recalculated based on:

1. The previous motivation level
2. The rate of impatience
3. Whether the sensory feedback indicates the behavior set is needed
4. Whether the robot has another behavior set already activated
5. Whether another robot has recently begun work on this task
6. Whether the robot is willing to give up the task, based on how long it has been attempting the task.

Effectively, the motivation continues to increase at some positive rate unless one of four situations occurs:

1. The sensory feedback indicates that the behavior set is no longer needed.
2. Another behavior set in the robot activates.
3. Some other robot has just taken over the task for the first time.
4. The robot has decided to acquiesce the task.

In any of these four situations, the motivation returns to zero. Otherwise, the motivation grows until it crosses a threshold value, at which time the behavior set is activated and the robot can be said to have selected an action. When an action is selected, cross-inhibition within that robot prevents other tasks from being activated within that same robot. When a behavior set is active in a robot, the robot broadcasts its current activity to other robots at a periodic rate.

The L-ALLIANCE extension [53.10] allows a robot to adapt the rate of change of the impatience and ac-

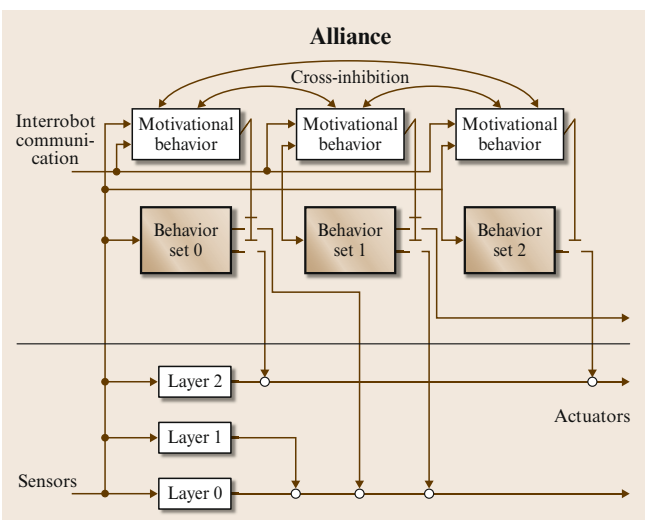


Fig. 53.2 The ALLIANCE architecture

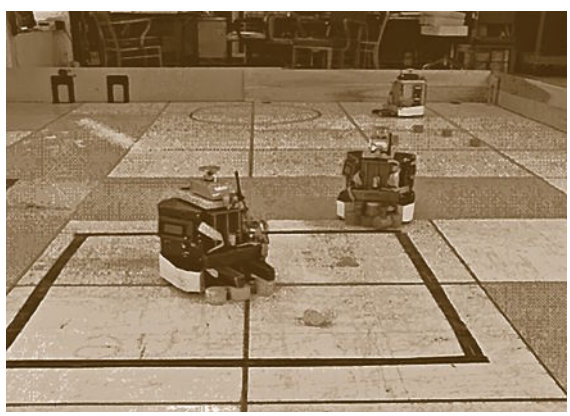


Fig. 53.3 Robots using the ALLIANCE architecture for a mock clean-up task

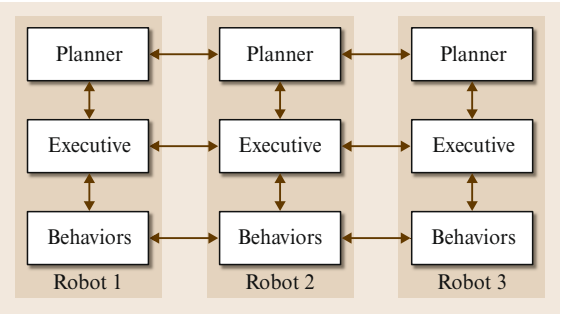


Fig. 53.4 The distributed robot architecture

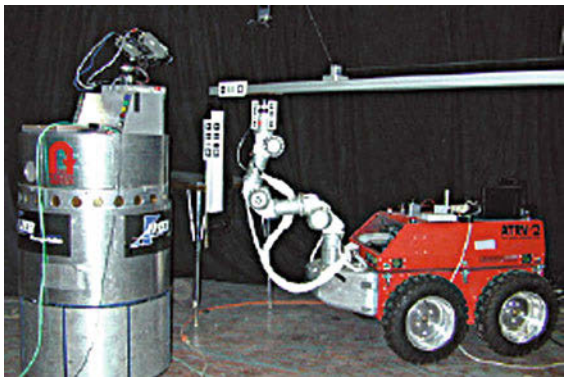


Fig. 53.5 Robots using the distributed robot architecture for assembly tasks

quiescence values depending on the quality with which that robot is expected to accomplish a given task. The result is that robots that have demonstrated their ability to better accomplish certain tasks are more likely to choose those tasks in the future. Additionally, if problems occur during team performance, then robots may dynamically reallocate their tasks to compensate for the problems. This approach was demonstrated on a team of three heterogeneous robots performing a mock clean-up task, two robots performing a box-pushing task, and four robots performing a cooperative target observation problem. The approach has also been demonstrated in the simulation of a janitorial service task and a bounding overwatch task. Figure 53.3 shows robots using ALLIANCE to perform the mock clean-up task.

53.2.3 The Distributed Robot Architecture

Simmons et al. [53.11] have developed a hybrid architecture called the distributed robot architecture (DIRA). Similar to the Nerd Herd and ALLIANCE approaches, the DIRA approach allows autonomy in individual robots. However, unlike the previous approaches, DIRA also facilitates explicit coordination among robots. This approach is based on layered architectures that are popular for single-robot systems (Chap. 12). In this approach (Fig. 53.4), each robot's control architecture consists of a *planning* layer that decides how to achieve high-level goals; an *executive* layer that synchronizes agents, sequences tasks, and monitors task execution; and a *behavioral* layer that interfaces to the robot's sensors and effectors. Each of these layers interacts with those above and below it. Additionally, robots can interact with each other via direct connections at each of the layers.

This architecture has been demonstrated in a team of three robots – a crane, a roving eye, and a mobile manipulator – performing a construction assembly task (Fig. 53.5). This task requires the robots to work together to connect a beam at a given location. In these demonstrations, a *foreman* agent decides which robot should move the beam at which times. Initially, the crane moves the beam to the vicinity of the emplacement based on encoder feedback. The foreman then sets up a behavioral loop between the roving eye and the crane robot to servo the beam closer to the point of emplacement. Once the beam is close enough, the foreman tasks the roving eye and the mobile manipulator to servo the arm to grasp the beam. After contact is made, the foreman tasks the roving eye and the mobile manipulator to coordinate to servo the beam to the emplacement point, thus completing the task.

53.3 Communication

A fundamental assumption in multirobot systems research is that globally coherent and efficient solutions can be achieved through the interaction of robots lacking complete global information. However, achieving these globally coherent solutions typically requires robots to obtain information about their teammates' states or actions. This information can be obtained in a number of ways; the three most common techniques are:

1. The use of implicit communication *through the world* (called *stigmergy*), in which robots sense the effects of teammate's actions through their effects on the world [53.6, 12–16]

2. Passive action recognition, in which robots use sensors to directly observe the actions of their teammates [53.17]
3. Explicit (intentional) communication, in which robots directly and intentionally communicate relevant information through some active means, such as radio [53.9, 18–21] – an area widely studied in the field of networked robot systems.

Each of these mechanisms for exchanging information between robots has its own advantages and disadvantages [53.22]. *Stigmergy* is appealing because of its simplicity and its lack of dependence upon explicit communications channels and protocols. However, it is limited by the extent to which a robot's perception of the world reflects the salient states of the mission the robot team must accomplish. *Passive action recognition* is appealing because it does not depend upon a limited-bandwidth, fallible communication mechanism. As with implicit cooperation, however, it is limited by the degree to which a robot can successfully interpret its sensory information, as well as the difficulty of analyzing the actions of robot team members. Finally, the *explicit communication* approach is appealing because of its directness and the ease with which robots can become aware of the actions and/or goals of its teammates. The major uses of explicit communication in multirobot teams are to synchronize actions, exchange information, and to negotiate between robots. Explicit communication is a way of dealing with the *hidden-state* problem [53.23], in which limited sensors cannot distinguish between different states of the world that are important for task performance. However, explicit communication is limited in terms of fault tolerance and reliability, because it typically depends upon a noisy, limited-bandwidth communications channel that may not continually connect all members of the robot team. Thus, approaches that make use of explicit communications must also provide mechanisms to handle communication failures and lost messages.

Selecting the appropriate use of communication in a multirobot team is a design choice dependent upon the tasks to be achieved by the multirobot team. One needs to carefully consider the costs and benefits of alternative communications approaches to determine the method that can reliably achieve the required level of system performance. Researchers generally agree that communication can have a strong positive impact on the performance of the team. One of the earliest illustrations of this impact was given in the work of *MacLennan* and *Burghardt* [53.24], which investigates the evolution of communication in simulated worlds and concludes that the communication of local robot information can result in significant performance improvements. Interestingly, for many representative applications, researchers have found a nonlinear relationship between the amount of information communicated and its impact on the performance of the team. Typically, even a small amount of information can have a significant impact on the team, as found in the study of *Balch* and *Arkin* [53.25]. However, more information does not necessarily continue to improve performance, as it can quickly overload the communications bandwidth without providing an application benefit. The challenge in multirobot systems is to discover the *optimal* pieces of information to exchange that yield these performance improvements without saturating the communications bandwidth. Currently, no general approaches to identifying this critical information are available; thus, the decision of what to communicate is an application-specific question to be answered by the system designer. *Dudek* et al.'s taxonomy of multirobot systems [53.26] includes axes related to communication, including communication range, communication topology, and communication bandwidth. These characteristics can be used to compare and contrast multirobot systems.

Several related issues of active research in communications for multirobot teams deal with dynamic network connectivity and topologies; for example, robot teams must either be able to maintain communications connectivity as they move, or employ recovery strategies that allow the robot team to recover when the communications connectivity is broken. These concerns may require robots to adapt their actions in response to the anticipated effects on the communications network, or in response to knowledge of the anticipated propagation behavior of information through the dynamic network. These and related issues are discussed next in the context of *networked robot systems*.

53.4 Networked Mobile Robots

Networked robots are multiple robots operating together coordinating and cooperating by *networked communication* to accomplish a specified task. Multiple robots enable new capabilities and the communication network enables new approaches and solutions that are difficult with just perception and control. Communication enables new control and perception capabilities in the system (e.g., access to information outside the perception range of the robot system). Conversely, control enables solutions for problems that are difficult with-

out mobility (e.g., localization). Section 53.4.1 defines the field, examines the benefits of networking in robot coordination, and discusses applications. Section 53.4.2 highlights a few projects focused on networked robotics and discusses the application potential of the field. Section 53.4.3 discusses the research challenges at the intersection of control, communication, and perception. Section 53.4.4 defines a model for the control of a networked system which is used in Sects. 53.4.5–53.4.8 to examine specific research issues and opportunities facilitated by the interplay between communication, control, and perception.

53.4.1 Overview

The term *networked robots* refers to multiple robots operating together coordinating and cooperating by *networked communication* to accomplish a specified task. Communication between entities is fundamental to cooperation (and coordination), hence there is a central role for the communication network in networked robots. Networked robots may also involve coordination and cooperation with stationary sensors, embedded computers, and human users. The central feature of networked robots is the ability of the system to perform tasks that are well beyond the abilities of a single robot or multiple uncoordinated robots.

The IEEE (Institute of Electrical and Electronics Engineers) *Technical Committee on Networked Robots* has adopted the following definition of a networked robot:

A networked robot is a robotic device connected to a communications network such as the Internet or local-area network (LAN). The network could be wired or wireless, and based on any of of a variety of protocols such as the transmission control protocol (TCP), the user datagram protocol (UDP), or 802.11. Many new applications are now being developed ranging from automation to exploration. There are two subclasses of networked robots:

1. Teleoperated, where human supervisors send commands and receive feedback via the network. Such systems support research, education, and public awareness by making valuable resources accessible to broad audiences.
2. Autonomous, where robots and sensors exchange data via the network. In such systems, the sensor network extends the effective sensing range of the robots, allowing them to communicate with each other over long distances to coordinate their activity. Sensing, actuation, and computation need no longer be collocated. A broad challenge is to develop a science base that couples communication, perception, and control to enable such new capabilities.

This definition of autonomous networked robots also includes a third class of distributed systems, mobile sensor networks, which is a natural evolution of sensor networks. Robot networks allow robots to measure spatially and temporally distributed phenomena more efficiently. The robots in turn can deploy, repair, and maintain the sensor network to increase its longevity, and utility. The focus of this chapter is *autonomous networked robots*.

Embedded computers and sensors are becoming ubiquitous in homes and factories, and increasingly wireless ad hoc networks or plug-and-play wired networks are becoming commonplace. Human users interact with embedded computers and sensors to perform tasks ranging from monitoring (e.g., supervising the operation of a factor and surveillance in a building) to control (e.g., running an assembly line consisting of sensors, actuators, and material-handling equipment). In most of these cases, the human users, embedded computers, and sensors are not collocated and the coordination and communication happens through a network. *Networked robots* extends this vision to multiple robots functioning in a wide range of environments performing tasks that require them to coordinate with other robots, cooperate with humans, and act on information derived from multiple sensors.

Figure 53.6 shows prototype concepts derived from academic laboratories and industry. In all these exam-

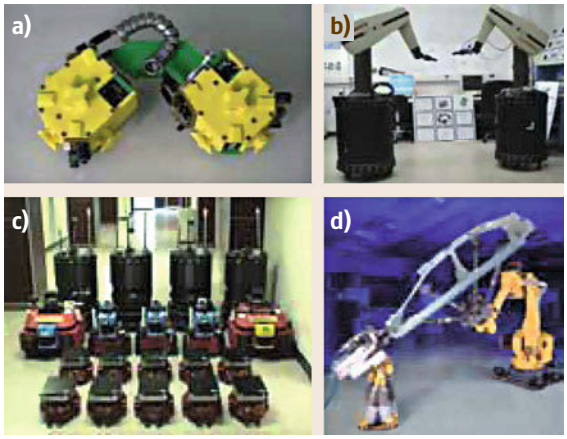


Fig.53.6a–d Small modules (after [53.27]) can automatically connect and communicate information to perform locomotion tasks **(a)**; robot arms (after [53.28]) on mobile bases can cooperate to perform household chores **(b)**; swarms of robots (after [53.29]) can be used to explore an unknown environment **(c)**; and industrial robots can cooperate in welding operations **(d)**

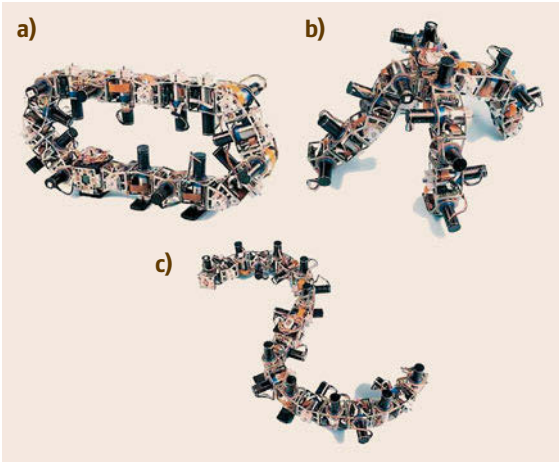


Fig.53.7a–c Robotic modules (after [53.30]) can be reconfigured to morph into different locomotion systems including **(a)** a wheel-like rolling system, **(b)** a snake-like undulatory locomotion system, and **(c)** a four-legged walking system

ples, independent robot or robotic modules can cooperate to perform tasks that a single robot (or module) cannot perform. Robots can automatically couple to perform locomotion tasks (also Fig. 53.7) and manipulation tasks that either a single robot cannot perform, or that would require a special-purpose larger robot to perform. They can also coordinate to perform search and reconnaissance tasks exploiting the efficiency that is inherent in parallelism. They can also perform independent tasks that need to be coordinated. Examples in the manufacturing industry include, for example, fixturing and welding.

Besides being able to perform tasks that individual robots cannot perform, networked robots also result in improved efficiency. Networking gives each robot access to information outside its perception range. Tasks such as searching or mapping can, in principle, be performed faster with an increase in the number of robots. A speed up in manufacturing operations can be achieved by deploying multiple robots performing operations in parallel but in a coordinated fashion.

Another advantage of using the network to connect robots is the ability to connect and harness physically removed assets. Mobile robots can react to information sensed by other mobile robots at a remote location. Industrial robots can adapt their end-effectors to new parts being manufactured upstream in the assembly line. Human users can use machines that are remotely located via the network.

The ability to network robots also enables fault tolerance in design. If robots can dynamically reconfigure themselves using the network, they are more tolerant to robot failures. This is seen in the Internet where multiple gateways, routers, and computers provide for a fault-tolerant system (although the Internet is not robust in other ways). Similarly, robots that can *plug* and *play* can be swapped in and out automatically to provide for a robust operating environment.

Finally, networked robots have the potential to provide great synergy by bringing together components with complementary benefits and making the whole greater than the sum of the parts.

Applications for networked robots abound. The US military routinely deploys unmanned vehicles that are reprogrammed remotely based on intelligence gathered by other unmanned vehicles, sometimes automatically. The deployment of satellites in space, often by astronauts in a shuttle with the shuttle robot arm, requires the coordination of complex instrumentation onboard the space shuttle, human operators on a ground station, the shuttle arm, and a human user on the shuttle. Home appliances now contain sensors and are becoming networked. As domestic and personal robots become more commonplace, it is natural to see these robots working with sensors and appliances in the house while cooperating with one or more human users. Networked robots will likely be used as critical ingredients in the environmental observatories of the future. Large-scale ecological monitoring precludes the use of monolithic infrastructure, and is envisioned to be built as a distributed, networked robotic system.

53.4.2 State of the Art and Potential

The growth in networked robot systems is broad-based, across many industries. There is a strong connection between this industry and the industry connected to sensor networks. Sensor networks have been projected to grow dramatically in terms of commercialization and market value [53.31]. Robot networks are analogous to sensor networks except that they allow sensors to have mobility and allow the geographical distribution of the sensors to be adapted based on the information acquired.

A system of robots, embedded computers, actuators, and sensors has tremendous potential in civilian, defense, and manufacturing applications. Nature provides the proof of concept of what is possible [53.32]. Group behaviors in nature can be found in organisms that are only microns to those that are several meters in length. There are numerous examples of simple animals that execute simple behaviors with modest sensors and actuators but communicate with and sense nearest neighbors to enable complex emergent behaviors that are fundamental to navigation, foraging, hunting, constructing nests, survival, and eventually growth. As seen



Fig. 53.8 Ants are able to cooperatively manipulate and transport objects often in large groups, without identified or labeled neighbors, and without centralized coordination

in Fig. 53.8, relatively small agents are able to manipulate objects that are significantly larger in terms of size and payload by cooperating with fairly simple individual behaviors. The coordination between agents is completely decentralized, allowing scaling up to large numbers of robots and large objects [53.33]. Individuals do not recognize each other. In other words, there is no labeling or identification of robots. The number of agents in the team is not explicitly encoded. Agents are identical, enabling robustness to failures and modularity. There is minimal communication, and even that which is present is only between neighbors. Furthermore, the optimal mode of group coordination may be scale dependent. Studies of wasps show strong evidence of centralized coordination among species with small colony sizes, but a distributed, decentralized coordination in larger colonies [53.34]. All these attributes are relevant to networked robots.

Biology has shown how simple decentralized behaviors in unidentified individuals (e.g., insects and birds exhibiting swarming behaviors) can exhibit a wide array of seemingly intelligent group behaviors. Similarly networked robots can potentially communicate and cooperate with each other, and even though individual robots may not be sophisticated, it is possible for networked robots to provide a range of intelligent behaviors that are beyond the scope of intelligent robots.

The significance and potential impact of networked robots is apparent from the following examples.

The manufacturing industry has always relied on integration between sensors, actuators, material-handling equipment, and robots. Today companies are finding it easier to reconfigure existing infrastructure by networking new robots and sensors with existing robots via wireless networks. There is also an increasing trend toward robots interacting with each other in operations like welding and machining, and robots cooperating with humans in assembly and material-handling tasks. Workcells consist of multiple robots, numerous sensors and controllers, automated guided vehicles, and one or two human operators working in a supervisory role. However, in most of these cells, the networked robots operate in a structured environment with very little variation in configuration and/or operating conditions.

There is a growing emphasis on networking robots in applications of field robotics, for example, in the mining industry. Like the manufacturing industry, operating conditions are often unpleasant and the tasks are repetitive. However, these applications are less structured and human operators play a more important role.

In the health care industry, networks allow health care professionals to interact with their patients, other professionals, expensive diagnostic instruments, and surgical robots. Telemedicine is expected to provide a major growth impetus for remote networked robotic devices that will take the place of today's stand-alone medical devices.

There are already many commercial products, notably in Japan, where robots can be programmed via and communicate with cellular phones. For example, the MARON robot developed by Fujitsu lets a human user dial up their robot and instruct it to conduct simple tasks including sending pictures back to the user via a cellular phone. Indeed these robots will interact with other sensors and actuators in the home – door openers equipped with Bluetooth cards and actuators and computer-controlled lighting, microwaves, and dishwashers. Indeed the Network Robot Forum [53.35] is already setting standards for how stationary sensors and actuators can interact with other robots in domestic and commercial settings.

Environmental monitoring is a key application for networked robots. By exploiting mobility and communication, robotic infrastructure enables observation and data collection at unprecedented scales in various aspects of ecological monitoring. This is significant for environmental regulatory policies (e.g., clean air and water legislation), as well as an enabler of new scientific discovery. For example, it is possible to obtain maps of salinity gradients in oceans, temperature and humidity variations in forests, and chemical composition of air and water in different ecological systems [53.36]. In addition to mobile sensor networks, it is also possible to use robots to deploy sensors and to retrieve information from the sensors. Mobile platforms allow the same sensor to collect data from multiple locations while communication allows the coordinated control and aggregation of information. Examples include systems built for aquatic [53.37], terrestrial [53.38], and subsoil monitoring [53.39]. There are many efforts to developed networked underwater platforms [53.40–42].

Networks of static and robotic devices have been developed for aquatic monitoring [53.37] and to obtain high-resolution information on the spatial and temporal distributions of plankton assemblages and concomitant environmental parameters. The RiverNet project [53.43] at Rensselaer Polytechnic Institute (RPI) has focused on the development of robotic sensor networks for monitoring a river ecosystem. Recent work at University of California, Los Angeles (UCLA), University of Southern California (USC), University of California, Riverside, and University of California, Merced on the networked infomechanical system project [53.38] has focused on the development of robotic networks for monitoring the forest canopy, with a view to providing data for modeling canopy and undercover growth. Networked robotic mini-rhizotrons [53.39] are being deployed in the forest to monitor root growth in the soil.

In the defense industry, countries like the USA have invested heavily in the concept of networked, geographically distributed assets. Unmanned aerial vehicles like the Predators are operated remotely. Information from sensors on the Predators triggers the deployment of other vehicles and weapon systems at a different remote location and allows commanders in a third location to control and command all these assets. The US military engaged in the large Future Combat Systems initiative to develop network-centric approaches to deploying autonomous vehicles. The network-centric tactical paradigms for modern warfare have created networked robots for defense and homeland security. While networked robots are already in operation, current approaches are limited to human users commanding a single vehicle or sensor system. However, it takes many human operators (between 2–10 depending on the complexity of the system) to deploy complex systems like unmanned aerial vehicles. A Predator unmanned aerial vehicle (UAV) is operated from a tactical control station, which may be on an aircraft carrier, with a basic crew of 3–10 operators.

The eventual goal, however, is to enable a single human user to deploy networks of unmanned aerial, ground, surface, and underwater vehicles. There have been several recent demonstrations of multirobot systems exploring urban environments [53.44, 45] and the interiors of buildings [53.46, 47] to detect and track intruders, and transmit all of the above information to a remote operator. These examples show that it is possible to deploy networked robots using an off-the-shelf 802.11b wireless network and have the team be remotely tasked and monitored by a single operator. An example of a project with heterogeneous vehicles in an urban setting is shown in Fig. 53.9. An example of a project with heterogeneous vehicles in an indoor setting is shown in Fig. 53.10 wherein robots map an environment and deploy themselves to form a sensor network to detect intruders.

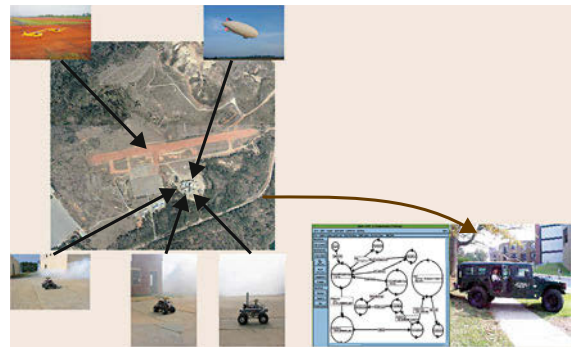


Fig. 53.9 A single operator commanding a network of aerial and ground vehicles from a command and control vehicle in an urban environment for scouting and reconnaissance in a recent demonstration by the University of Pennsylvania, Georgia Tech. and University of Southern California (after [53.48])

Many research projects are addressing group behaviors or collective intelligence by realizing swarming behaviors observed in nature. For example, the European Union (EU) funded several EU-wide coordinated projects on collective intelligence or swarm intelligence. The I-Swarm project in Karlsruhe [53.49] and the Swarm-Bot project at Ecole Polytechnique Fédérale de Lausanne (EPFL) [53.50] are examples of swarm intelligence. The Laboratory for Analysis and Architecture of Systems (LAAS) has a strong group in robotics and artificial intelligence. This group has had a long history of basic and applied research in multirobot systems. The integration of multiple unmanned vehicles for applications such as terrain mapping and fire-fighting is addressed in [53.51]. A multi-university US project addressed the development of networked vehicles for swarming behaviors [53.52]. Projects such as these are exploring the scalability of the basic concepts to large numbers of robots, sensors, and actuators.

53.4.3 Research Challenges

While there are many successful embodiments of networked robots with applications to manufacturing industry, the defense industry, space exploration, domestic assistance, and civilian infrastructure, there are significant challenges that have to be overcome.

The problem of coordinating multiple autonomous units and making them cooperate creates problems at the intersection of communication, control, and perception. Who should talk to whom and what information should be conveyed, and how? How does each unit



Fig. 53.10 Under the DARPA SDR program, a team from the University of Southern California, the University of Tennessee, and Science Applications and International Corporation (SAIC) demonstrated mapping, and intruder detection by a team of networked robots (after [53.46])

move in order to accomplish the task? How should the team members acquire information? How should the team aggregate information? These are all basic questions that need basic advances in control theory, perception, and networking. In addition, because humans are part of the network (as in the case of the Internet), we have to devise an effective way for multiple humans to be embedded in the network and command/control/monitor the network without worrying about the specificity of individual robots in the network. Thus the underlying research challenges lie at the intersection of control theory, perception, and communication/networks, as shown in Fig. 53.11.

It is also worth noting that robot networks are dynamic, unlike networks of sensors, computers or machines which might be networked together in a fixed topology. When a robot moves, its neighbors change and its relationship to the environment changes. As a consequence, the information it acquires and the actions it executes must change. Not only is the network topology dynamic, but the robot's behavior also changes as the topology changes. It is very difficult to predict the performance of such dynamic robot networks, yet it is this analysis problem that designers of robot networks must solve before deploying the network.

This notion of a changing topology inevitably leads us to complicated mathematical models. Traditionally, models of group behavior have been built on continuous models of dynamics of individuals, including local interactions with neighbors, and models of control and sensing with a fixed set of neighbors. While dynamics at the level of individual units may be adequately described by differential equations, the interactions with neighbors are best described by edges on a graph. Modeling, analysis, and control of such systems will require a comprehensive theoretical framework and new representational tools. New mathematical tools that marry dynamical system theory, switched systems, discrete mathematics, graph theory, and computational geometry are needed to solve the underlying problems. We need a design methodology for solving the *inverse*

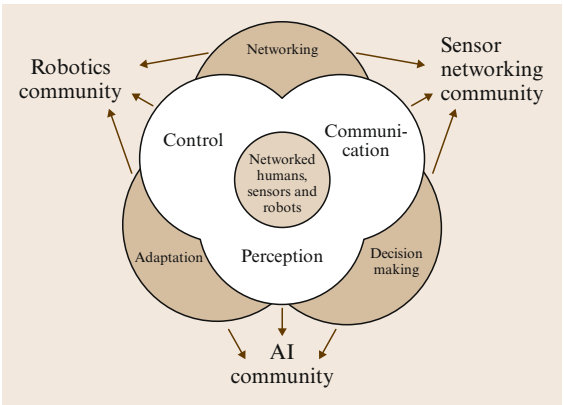


Fig. 53.11 The paradigm of *networked robots* introduces fundamental challenges at the intersection of control, perception, and communication that is of interest to the robotics, sensor networks, and artificial intelligence communities

problem in navigation – behaviors for controlling individuals to achieve a specified aggregate *motion* and *shape* of the group, and the application to active perception and coverage. An overview of some of these methods is provided in Sect. 53.4.4.

Problems of perception have been studied extensively in the robotics community. However, the perception problems in a system of networked, mobile sensor platforms bring a new set of challenges; for example, consider the problem of estimating the state of the network. State estimation requires the estimation of the state of robots and the environment based on local, limited-range sensory information. Localization of n vehicles in an m-dimensional configuration space requires $O((nm)^k)$ computations, where k is somewhere between 3 and 6, depending on the algorithm and domain-specific assumptions. The estimation problem is further exacerbated by the fact that not all robots in the network may be able to get the necessary information in a time-critical fashion. There are deep issues of representation and algorithmic development, which are discussed in Sect. 53.4.6.

The paradigm of active perception [53.53] links the control of sensor platforms to perception, bringing control theory and perception together in a common framework. Extending this paradigm to networked robots requires approaches of distributed control to be merged with decentralized estimation. Robots can move in order to localize themselves with respect to their neighbors, to localize their neighbors, and also to identify, localize, and track features in the environment. These problems are discussed in Sect. 53.4.7.

As discussed earlier, the communication network is central to the functioning of a network of robots.

However, if the network consists of mobile agents with transmitters and receivers with finite power, there is no guarantee that all agents can talk to teach other. Unlike a static sensor network, robots in a network can move toward each other to facilitate communication and adaptively maintain a communication network. Some basic algorithmic problems and several pertinent results are provided in Sect. 53.4.8.

53.4.4 Control

The control of individual robots is critical to the performance and scope of robot networks. Indeed motion coordination algorithms have been proposed for the purpose of improving communication performance [53.54, 55], localization [53.56, 57], information integration, deployment [53.58], and coverage [53.59–61], among other tasks. Mobility allows the group of robots to self-deploy, and self-organize by relocating themselves in support of communication, sensing, or task needs; for example, they can reconfigure to guarantee a desired communication bandwidth, k-hop connectivity, or algebraic connectivity, enabling message delivery from one robot to another. The group can also self-organize to position sensors so as to cover a desired area and adapt to shifts in the focus of monitoring activities. Controlling sensor position also supports map making, tracking of objects and events, and goal-directed navigation for users of the network. Finally, mobility allows robots to accomplish tasks such as navigation, reconnaissance, transportation, and search and rescue.

Given a group of mobile sensors, we would like to have distributed control capabilities that realize desirable global specifications. Thus, it is necessary to be able to automatically determine the necessary position and orientation of the group members and/or the distribution of group members, and their motion to achieve the desired task. At a lower level, the robots must be able to use information from the communication network and from their own sensors to derive local estimates, reason about the spatial network (their neighbors and their relationship to the environment), and then use the appropriate control policies to achieve the desired group specifications. We briefly outline the simplest mathematical model that is necessary to formulate such problems in order to provide a better sense of the underlying challenges.

In a robot network, we have multiple agents or nodes in which each agent is a physical entity that can be a robot, a vehicle with actuators and sensors, a sensor platform (possibly static) or even a communication relay node. Each agent A_i is characterized by an identifier, $i \in I \subset Z$, a state $x_i \in X_i \subset \mathbb{R}^n$, and control inputs $u_i \in U_i \subset \mathbb{R}$, with $f_i : X_i \times U_i \to TX_i$ specifying the dynamics

$$\dot{x}_i = f_i(x_i, u_i) \,. \tag{53.1}$$

The state x_i will consist of the position (and orientation), r_i in some d-dimensional space, and its velocity, $\dot{r}_i : x_i = \left(r_i^{\mathrm{T}}, \dot{r}_i^{\mathrm{T}}\right)^{\mathrm{T}}$, with $n = 2d$. $\mathcal{N}^{\mathrm{c}}(r_i)$ and $\mathcal{N}^{\mathrm{s}}(r_i)$ are neighborhoods of r that define the range and field of view of the communication hardware and sensors, respectively.

A network of robots S consists of N agents with a *sensing graph* and a *communications graph* that is defined by the physical distribution of the agents. The sensing graph (and similarly the communications graph) is defined by a map $E^{\mathrm{s}} : X^1 \times X^2 \cdots X^N \to I \times I$, where the edges of the graph are formed dynamically depending on the physical proximity of pairs of agents. Specifically, the $N \times N$ adjacency matrix, $\mathcal{A}^{\mathrm{s}}$ (and similarly $\mathcal{A}^{\mathrm{c}}$) has entries

$$\mathcal{A}^{\mathrm{s}}{}_{ij} = \begin{cases} 1, & \text{if } r_j \in \mathcal{N}^{\mathrm{s}}(r_i) \,, \\ 0, & \text{otherwise} \,. \end{cases} \tag{53.2}$$

Agent A_i has estimates of its own state and the states of neighbors (e.g., A_j), and these estimates are derived from information associated with edges in the sensing and communication graph

$$\hat{x}_j^{(i)} = h(x_i, z_{ij}) \,, \tag{53.3}$$

where z_{ij} represents measurements of the state of agent A_j available to A_i by sensing or communication channels and h is the estimator used by A_i. Note that z_{ij} may have dimension less than n and may therefore not contain complete information about $x_{ij} = x_i - x_j$. Clearly the relative position vector denoted by $r_{ij} = r_i - r_j$ and its magnitude are important quantities that may need to be estimated for biological and artificial agents.

Finally, A_i can encode n_{b_i} behaviors, which we will denote by $\mathcal{B}_i = B_1, B_2, \ldots, B_{n_{b_i}}$. Each behavior B_j is a controller, a function $k_j : \mathbb{R} \times X_i \to U_i$. All agents can be assigned identical or different behaviors. Each behavior represents a set of unsynchronized, locally executed computations (for control or estimation) being carried out for some collective purpose, with each processor using in its computations only data from its *neighboring* processors. Furthermore, even for a fixed assignment of behaviors, each processor's neighbors typically *change* with time because the processors are moving in and out of the sets $\mathcal{N}^{\mathrm{c}}$ and $\mathcal{N}^{\mathrm{s}}$. Thus the methodology for modeling and analyzing such systems will require the merging of graph theory and dynamical system theory at a fundamental level.

The reader is directed to the many survey articles on this subject for further information. An overview of challenges for the controls community is presented in [53.62]. The underlying theory for networked mobile systems has been explored in the context of automated highway systems [53.63], cooperative robot reconnaissance [53.46] and manipulation [53.64], formation flight control [53.65], and the control of groups of unmanned vehicles [53.45]. Our goal in the following sections is to explore the connections between communication, perception, and control.

53.4.5 Communication for Control

Communication networks allow physically disconnected entities to exchange information. At the lowest level, when groups of vehicles coordinate their actions, communication allows vehicles to exchange state information [53.66–68]. At a higher level, robots can plan navigation and exploration tasks based on an integrated map of the world derived from information acquired from different robots [53.52].

The use of communication for control in the multivehicle context has been addressed in the PATH project where formations of inline vehicles were studied [53.63]. Problems of the stability of the formation [53.69], the convergence of the formation to shapes [53.70], and the overall performance of the system [53.71] are of great interest. The performance of the system is directly influenced by the interconnections between agents. In addition to impacting on stability [53.63], feedback of states from different agents and feedforward information from the plans of different agents affects the rates at which the system of agents can respond to external stimuli [53.71] or to commands from human operators [53.72].

In addition, communication can be used for high-level control and planning of robots. There is great interest in using static sensor nodes as beacons to guide robot navigation. In [53.73], the problem of coverage and exploration of an unknown dynamic environment using a mobile robot is considered. An algorithm is presented which assumes that global information is not available (neither a map, nor global positioning system (GPS) information). The algorithm deploys a network of radio beacons that assists the robot in coverage. The network is also used by the robot for navigation. The deployed network can also be used for applications other than coverage (such as multirobot task allocation). A similar idea was presented using potential-field-based navigation in [53.52]. In this work the notion of no-go or danger areas was incorporated into the navigation cost function. Recent work along these lines with experimental data from sensor nodes is reported in [53.74].

In such communication-enabled cooperative control and planning (see also [53.75]), the communication network plays an important role in the creation of a shared representation of information. This notion of a shared representation is important to the scaling of coordinated control algorithms to large numbers of devices. For example, in [53.67], the information form of the Kalman filter is used to derive a framework for decentralizing estimation and fusion algorithms. This approach was shown to be applicable to multiple heterogenous ground and aerial platforms [53.56]. Such methodologies are transparent to the specificity and identity of the cooperating vehicles. This is because vehicles share a common representation, which consists of a certainty grid that contains information about the probability of detection of targets, and an information vector–matrix pair that is used in the information form of the Kalman filter [53.45]. Observations are propagated through the network by changing both the certainty grid and the information vector/matrix. This allows each vehicle to choose the action that maximizes a utility function, which is the combined mutual information gain from onboard sensors towards the detection and localization of features in the environment.

Thus, in summary, at the lowest level, communication enables either partial or complete state feedback of the network and allows agents to exchange information for feedforward control. At the higher levels, agents can share information for planning and for control. This is also discussed in Sect. 53.4.6 where the communication network is shown to enable a network-centric approach to perception.

53.4.6 Communication for Perception

While individual robots have sensors and the ability to build maps and models by integrating sensory information, networked robots can exchange information and leverage sensory data, maps, and models from other robots. The challenge is to exploit communication for perception in tasks such as distributed mapping in the presence of the delays, limited bandwidth, and disruption that are typical of communication networks.

Distributed localization is the term used to describe the merging of communication and perception for state estimation. Localization is an essential tool for the development of low-cost robot networks for use in location-aware applications and ubiquitous networking [53.76]. Location information is needed to track the placement of the nodes and to correlate the values measured by the node with their physical location. Distributed computation and robustness in the presence of measurement noise are key ingredients for a practical

localization algorithm that will give reliable results over a large-scale network.

The methods for distributed localization can be classified into two broad classes: algorithms that rely on anchor nodes for localization and algorithms that use no beacons. Localization may be computed using range information between nodes, bearing information, or both.

In [53.54] a theoretical foundation for network localization in terms of graph rigidity theory is provided. The problem is solved when nodes have perfect range information and it is show that a network has a unique localization if and only if its underlying graph is *generically globally rigid*. In [53.77] the Cramér–Rao lower bound (CRLB) for network localization is derived. This work computes the expected error characteristics for an ideal algorithm, and compares this to the actual error in an algorithm based on multilateration, drawing the important conclusion that the error introduced by the algorithm is just as important as the measurement error in assessing end-to-end localization accuracy. In [53.78] a distributed algorithm that uses no beacons and is guaranteed to compute correct location information under measurement noise for nodes that can range to neighbors is presented. This algorithm relies on the notion of robust quadrilaterals to compute robustly a global system of coordinates among the nodes. The computation supports moving nodes. Extensions of this work to passive tracking have been discussed in [53.79]. Localization based on the propagation of location information from known reference nodes based on connectivity includes [53.80, 81]. Mobility-assisted localization is introduced in [53.82]. Other techniques use distributed propagation of location information using multilateration [53.77, 83]. In a recent paper [53.84] the problem of evaluating the rigidity of a planar network is treated while satisfying common objectives of real-world systems: decentralization, asynchronicity, and parallelization.

Two approaches for cooperative relative localization of mobile robot teams are given in [53.85, 86]. Neither method uses GPS, landmarks, or maps of any kind; instead, robots make direct measurements of the relative pose of nearby robots and broadcast this information to the team as a whole. In [53.85], each robot processes this information independently to generate an egocentric estimate for the pose of other robots using a Bayesian formalism with a particle filter implementation. In [53.86], maximum-likelihood estimation (MLE) and numerical optimization is used to achieve a similar result.

A key issue is to be able to scale these computations for building a shared representation to large numbers of robots and sensors. This problem was studied in experiments under the US Defense Advanced Research Projects Agency (DARPA-funded software for distributed robotics (SDR) program. The goal of these experiments was to develop and demonstrate a multi-robot system capable of carrying out a specific mission. This required the ability to deploy a large number of robots into an unexplored building, map the building interior, detect and track intruders, and transmit all of the above information to a remote operator. A report on one set of experiments is presented in [53.46]. A tiered strategy for deploying the robots is described, where highly capable robots formed the first wave to enter and map a building, followed by a second wave which used the resulting map to self-deploy and monitor the environment for intruders. Both approaches relied extensively on networking the robots using commercial 802.11b wireless technology. This task involved both communication for building a shared representation as well control for perception.

Another important set of problems arises when robot networks are used for identifying, localizing, and then tracking targets in a dynamic setting. An embedded stationary wireless sensor network is like a virtual sensor spread over a large geographical area. Such a network can provide information to mobile robots about remote locations. Robot networks allow this virtual sensor to move in response to external stimuli and to track moving targets. Indeed, it is possible to cast this scenario as a pursuit-evasion game with robotic sensor networks [53.87]. For example, the Tenet project at USC addresses the design of network primitives and abstractions for tiered network architectures, with robotic pursuit evasion as one of the target applications. Algorithms for guiding the sampling strategy of a robotic boat to model and locate phenomena of interest (e.g., hotspots) in aquatic environments are discussed in [53.37]. The networked infomechanical systems (NIMS) project has focused on sensor-assisted techniques for mobile robot-based adaptive sampling for event response [53.88] and field reconstruction [53.89].

The information collected by the nodes in a sensor network can be processed at a central location or in a decentralized fashion. Such in-network data processing techniques make better use of network communication and computation resources than centralized processing. This also enables the network to compute accurate and up-to-date global pictures of the global perception landscape that are available to all the robots in the system. Methods for in-network data processing with static nodes include artificial potential-field computation, gradient computations, particle filters, Bayesian inference, and signal processing. Algorithms have been developed for computing maps, paths, and predictors [53.52, 73, 90].

A recent DARPA demonstration showed how communication networks can be used effectively in perception tasks involving heterogenous robots [53.44]. In cooperative search, identification, and localization unmanned aerial vehicles (UAVs) can be used to cover large areas, searching for targets. However, sensors on UAVs are typically limited in their accuracy of localization of targets on the ground. On the other hand, ground robots can be deployed to accurately locate ground targets but have the disadvantage of not being able to move rapidly and see through obstacles such as buildings or fences. In [53.56], the synergy between these two devices is exploited by creating a seamless network of UAVs and unmanned ground vehicles (UGVs). As discussed in Sect. 53.2, the key to such network-centric approaches for search and localization is a shared representation of state information, which in this case is easily scalable to large numbers of UAVs and UGVs and is transparent to the specificity of individual platforms. However, how to do this more generally and for more unstructured information remains an issue for future research.

53.4.7 Control for Perception

Networked mobile robots enable the exploration of dynamic environments and the recovery of three-dimensional information via distributed active perception [53.53]. Since the nodes are mobile, a natural question is: where should the nodes be placed in order to ensure successful integration of information from multiple nodes, and to maximize the quality of the estimates returned by the team? Since there is a cost associated with transmitting and processing data, it is important to consider which sensor readings should be used in the state estimation and what information should be communicated to the rest of the system. The quality of the information computed by the network depends on the locations of the sensor platforms both in an absolute and relative sense. The quality also depends on the noise characteristics of each sensor, and the communication network.

A robot network goes well beyond a fixed sensor network, which can only collect data at fixed positions in space; for example, when an event is detected at a specific location it is possible to direct more sensors toward the location of observation of the event for more information (for example, higher-resolution data or higher sampling frequency). Reconfiguring the node locations for adaptive resolution sampling relies on distributed control strategies.

Various strategies have been introduced for controlling mobile sensor network coverage. Mobile sensing agents are controlled using gradients of information-based objective functions [53.91]. Stability results are derived without concerns for the optimality of the network configuration, but local guarantees are provided. Topology aware coordinated behavior is treated in [53.92]. A body of results reported in [53.93] and [53.94] describes decentralized control laws for positioning mobile sensor networks optimally with respect to a known event distribution density function. This approach is advantageous because it guarantees that the network (locally) minimizes a cost function relevant to the coverage problem. However, the control strategy requires that each agent have a complete knowledge of the event distribution density, thus it is not reactive to the sensed environment. The work by [53.95, 96] generalizes these results to situations in which the nodes estimate rather than know ahead of time the event distribution density function. A local (decentralized) control law requires that each agent can measure the value and gradient of the distribution density function at its own position. This results in a sensor network that is reactive to its sensed environment while maintaining or seeking a near-optimal sensing configuration. In addition, the distribution density function approximation yields a closed-form expression for the control law in terms of the vertices of an agent's Voronoi region. This eliminates the need for the numerical integration of a function over a polynomial domain at every time step, thereby providing a significant reduction in computational overhead for each agent. Other work in event monitoring for unknown distributions includes [53.59]. *Krause* et al. [53.97] have recently proposed an approach for sensor placement that considers both the sensing quality and communication cost of imperfect sensing and communication components. They use a parametric model for link reception rate that assumes no acknowledgement and no temporal correlation of lossy links.

Beginning with the art gallery problem, there have been multiple efforts to determine an optimal configuration of sensors to cover a given region [53.98–100]. A variant which allows the use of mobile sensors is known as the *watchmen tours* problem. In these approaches the sensor model is abstract and not well suited to real environments and cameras. Distributed geometric optimization methods [53.94] have also been used for mobile sensor network reconfiguration. A related class of methods is the use of estimation-theoretic optimization metrics and the application of information filters to coordinate network-wide motion [53.56]. There are other distributed optimization methods which use a distributed control law and show that it optimizes a global metric of interest, such as using a potential field or other linear control law based only on local neighbor interactions [53.101]. Research focusing on

the control of cameras with pan, tilt, and zoom capabilities is due to [53.60, 102, 103]. In [53.102] an approach is developed to calibrate a pan–tilt–zoom camera automatically over its full zoom range and to build very high-resolution panoramas. In [53.60], the cameras are constantly moved to track observed targets, using a factor graph. A recent algorithm due to [53.104] significantly improves on this by positioning cameras to make the network better suited to detect and classify targets as they emerge. Pan–tilt–zoom cameras allow the construction of far more flexible vision systems than static cameras.

53.4.8 Control for Communication

In Sect. 53.4.5, we briefly discussed the benefits of using the communication network to synthesize and improve controller design. Conversely, the movement of robots affects the network and data transmission in the network. This gives rise to many challenges. If the controllers for individual robots are known, can we provide guarantees about communication in the network and can we develop robust information routing and networking algorithms in the presence of robot motion? Another challenge concerns how information propagates and diffuses in these networks. If the robots move under a given control model, how does a piece of information propagate through the network and what can we say about when and where that piece of information will be heard? If we know the answers to such questions, it may be possible to design controllers to realize desired communication network characteristics.

One simple control strategy that can affect network performance is to control the robot motion to ensure messages are transmitted between designated nodes. The movement of robots in a network of robots and sensors may cause network partitioning when nodes go out of range. However, the ability of the robots to move in a controlled way also leads to an opportunity to address the information routing problem in disconnected networks by turning the robots into relay nodes. The key idea here is to enable the robot holding a current message to an unavailable destination to modify their trajectory in order to relay a message. This problem has been formulated as an optimization problem. The goal is to minimize the trajectory modifications necessary to send a message to its destination. Several solutions have been proposed depending on the information that is available to the robots. If the robots' trajectories are known, path planning techniques can be used to compute who moves where to relay what. If the robots' trajectories are not known, a distributed spanning tree can be created to enable robots to keep track of each other. Each robot is assigned a region of movement and a parent in the spanning tree. When the robot leaves its region, the parent is informed. When the robot moves too far away, the spanning tree is modified.

Mobile robots can be used to create desired network topologies under suitable models of network communication. If a robot is used to emplace nodes in an environment (or if sensor nodes robotically self-deploy) to build a network, the problem is referred to as deployment. It is possible to control the motion of individual nodes to guarantee that a specified topology is maintained [53.55]. It is also possible to reposition nodes with the explicit aim of changing the network topology – the so-called mobility-based topology control problem.

A distributed algorithm for the deployment of mobile robot teams has been described by [53.105] using the concept of virtual pheromones: localized messages from one robot to another. These messages are used to generate either a gas expansion or a guided growth deployment model. Similar algorithms based on artificial potential fields are described in [53.106, 107], where the latter incorporates a connectivity constraint. An incremental deployment algorithm for mobile sensor networks is given in [53.58]; nodes are deployed one at a time into an unknown environment, with each node making use of information gathered by previously deployed nodes to determine its deployment location. The algorithm is designed to maximize network coverage while ensuring that nodes retain line of sight with one another.

Most work on network topology control has dealt with uncontrolled deployments, where there is no explicit control of the positions of individual nodes. The primary mechanisms proposed are power control and sleep scheduling. These methods involve pruning an already existing, well-connected communication graph in order to save power while ensuring that the resultant subgraph preserves connectivity. Given a network that is connected when all nodes are operating at maximum power, the aim of power control is to use the minimum power level at each node for which the network remains connected [53.108]. Given an overdeployed network, sleep scheduling seeks to activate a minimal subset of nodes to maintain connectivity and achieve other desired metrics [53.109]. In contrast, controlled deployments are feasible when the positions of individual nodes can be altered. Such deployments are interesting for two reasons. First, network topology with wireless communication relates directly to proximity relations and hence the position of the nodes. Second, there is increasing evidence that a large number of deployments are likely to involve careful, nonrandom placement of nodes. The positioning of nodes is controlled either by the nodes themselves or by external agents. Such net-

works present a different and interesting scenario for topology control since it is possible to exploit control of the motion and placement of the nodes to build efficient topologies. A local, completely decentralized technique for topology control using mobility is given in [53.110].

An important application for networked robots is in monitoring and surveillance, where it is important that the robots cover the space while remaining within communication range [53.111–113]. In a recent development [53.114] the problem of how to design communication models and scheduling protocols for choosing the appropriate path planning algorithms for robotic data collection is discussed. Probing environment and adaptive sleeping protocol (PEAS) was one of the first attempts to address communication connectivity and sensing coverage simultaneously using heuristic algorithms [53.115]. *Wang* et al. [53.116] proposed a new coverage configuration protocol (CCP) to produce an approach that simultaneously optimizes coverage and connectivity while maximizing the number of nodes that are placed into sleep mode. Furthermore, they also identified three different classes of coverage–connectivity problems with respect to the ratio of radio and sensing ranges and recognized the critical ratio where the former range is twice as long as the latter. *Zhang* and *Hou* proved that, if the communication range is at least twice the sensing range, complete coverage of a convex area guarantees network communication connectivity, and then used this theorem as a basis for a localized density control algorithm [53.109]. This was subsequently generalized to show that the condition that the communication range is twice the sensing range is sufficient and is the tight lower bound to guarantee that complete coverage preservation implies communication connectivity among nodes if the original network topology is connected [53.117].

In summary, if the state of the communication network and the desired state of the communication network is known to each agent, it should be possible to synthesize distributed controllers to move agents to attain desired network characteristics. However, the assumptions on the global state are clearly not justified. Also, the desired motion to optimize network characteristics will conflict with the motion that is required to perform the desired task. However, as the brief discussion above illustrates, there are many interesting studies that point to promising directions for future work in this very fertile research field.

53.5 Swarm Robots

Historically, some of the earliest work in multirobot systems [53.12, 13, 118–125] dealt with large numbers of homogeneous robots, called *swarms*; swarm robotics continues to be a very active area of research. Most swarm approaches obtain inspiration from biological societies – particularly ants, bees, and birds – to develop similar behaviors in multirobot teams. Because biological societies are able to accomplish impressive group capabilities, such as the ability of termites to build large complex mounds, or the ability of ants to collectively carry large prey, robotics researchers aim to reproduce these capabilities in robot societies.

Swarm robotics systems are often called *collective* robotics, indicating that individual robots are often unaware of the actions of other robots in the system, other than information on proximity. These approaches aim to achieve a desired team-level global behavior from the interaction dynamics of individual robots following relatively simple local control laws. Swarm robotic systems typically involve very little explicit communication between robots, and instead rely on stigmergy (i. e., communication through the world) to achieve emergent cooperation. Individual robots are assumed to have minimal capabilities, with little ability to solve meaningful tasks on their own. However, when grouped with other similar robots, they are collectively able to achieve team-level tasks. Ideally, the entire team should be able to achieve much more than individual robots working alone (i. e., it is *superadditive*, meaning that the whole is bigger than the sum of the parts). These systems assume very large numbers of robots (at least dozens, and often hundreds or thousands) and explicitly address issues of scalability. Swarm robotic approaches achieve high levels of redundancy because robots are assumed to be identical, and thus interchangeable with each other.

Many types of swarm behaviors have been studied, such as foraging, flocking, chaining, search, herding, aggregation, and containment. The majority of these swarm behaviors deal with spatially distributed multirobot motions, requiring robots to coordinate motions either:

1. Relative to other robots
2. Relative to the environment
3. Relative to external agents
4. Relative to robots and the environment
5. Relative to all (i. e., other robots, external agents, and the environment).

Table 53.1 categorizes swarm robot behaviors according to these groupings, citing representative examples of relevant research.

Table 53.1 Categories of swarm behaviors

Relative motion requirements	Swarm behaviors
Relative to other robots	Formations [53.126, 127], flocking [53.121], natural herding (as in herds of cattle), schooling, sorting [53.14], clumping [53.14], condensation, aggregation [53.128], dispersion [53.129]
Relative to the environment	Search [53.130], foraging [53.131], grazing, harvesting, deployment [53.58], coverage [53.132], localization [53.133], mapping [53.134], exploration [53.135]
Relative to external agents	Pursuit [53.136], predator–prey [53.137], target tracking [53.138], forced herding/shepherding (as in shepherding sheep)
Relative to other robots and the environment	Containment, orbiting, surrounding, perimeter search [53.139]
Relative to other robots, external agents, and the environment	Evasion, tactical overwatch, soccer [53.140]

Much of the current research in swarm robotics is aimed at developing specific solutions to one or more of the swarm behaviors listed in Table 53.1 (VIDEO 214). Some of these swarm behaviors have received particular attention, notably formations, flocking, search, coverage, and foraging. Section 53.10 discusses these behaviors in more detail. In general, most current work in the development of swarm behaviors is aimed not just at demonstrating group motions that are similar to biological systems, but also at understanding the formal control theoretic principles that can predictably converge to the desired group behaviors, and remain in stable states.

Demonstration of physical robot swarms is both a hardware and a software challenge. As discussed in Sect. 53.2, the first demonstrations were by *Matarić* [53.6], involving about 20 physical robots performing aggregation, dispersion, and flocking. This work defined composable *basis behaviors* as primitives for structuring more complex systems. In later research, *McLurkin* [53.141] (VIDEO 215) developed an extensive catalog of swarm behavior software, and demonstrated these behaviors on about 100 physical robots (called the SwarmBot robots), developed by iRobot, as shown in Fig. 53.12. He created several group behaviors, such as avoidManyRobots, disperseFromSource, disperseFromLeaves, disperseUniformly, computeAverageBearing, avoidManyRobots, followTheLeader, orbitGroup, navigateGradient, clusterOnSource, and clusterIntoGroups. A swarm of 108 robots used the developed dispersion algorithms in an empty schoolhouse of area of about 300 m^2, and were able to locate an object of interest and lead a human to its location [53.129].

The European Union has sponsored many swarm robot projects, leading toward decreasingly smaller-sized individual robots, and increasingly larger-sized robot teams. The I-SWARM project, for instance, aimed at developing millimeter-sized robots with full onboard sensing, computation, and power for performing biologically inspired swarming behaviors, as well as collective perception tasks. This project was both a hardware and a software challenge, in that developing microscale robots that are fully autonomous and can perform meaningful cooperative behaviors will require significant advances in the current state of the art.

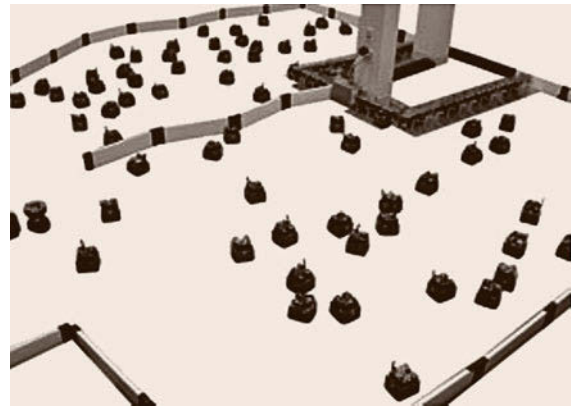


Fig. 53.12 The SwarmBot robots

The EU-based SWARM-BOTS project studied new concepts in the design and implementation of self-organizing and self-assembling robots (VIDEO 195). In this work [53.142], *s-bot* robots are developed that have grippers enabling the robots to create physical links with other s-bots or objects, thus creating assemblies of robots. These assemblies can then work together for navigation across rough terrain, or to collectively transport objects. The s-bots are cylindrical, with a flexible arm and toothed gripper that can connect one s-bot to another. An interesting application of object transport using SWARM-BOTs is shown in Figure 53.13, which shows the robots self-assembling into four chains in order to pull a child across the floor (VIDEO 212).

Another notable effort in swarm robotics research is the US multi-university SWARMS initiative led by the University of Pennsylvania. Research in this project aimed at developing a new system-theoretic frame-



Fig. 53.13 SWARM-BOTs self-assembling to move a child across the floor

work for swarming, developing models of swarms and swarming behavior, analyzing swarm formation, stability, and robustness, synthesizing emergent behaviors for active perception and coverage, and developing algorithms for distributed localization.

Besides the hardware challenges of dealing with large numbers of small robots, there are many important software challenges that remain to be solved. From a practical perspective, a common approach to creating homogeneous multirobot swarms is to hypothesize a possible local control law (or laws), and then study the resulting group behavior, iterating until the desired global behavior is obtained. However, the longer-term objective is to be able to both predict group performance based on known local control laws, and to generate local control laws based upon a desired global group behavior.

More recent research has focused on the development of analytical techniques that can synthesize distributed controllers that achieve the desired macro-level system behaviors. *Mather* and *Hsieh* [53.143] address this challenge by proposing a technique that first identifies robot–robot interactions at the macroscopic level; they then use this analysis to improve local robot control policies by filtering out spurious robot–robot interactions. Another top-down design approach is presented by *Chen* et al. [53.144] who show how to automatically synthesize control and communication strategies for a robot team based on global specifications of the desired system-level behavior, stated using regular expressions. The resulting control strategies are formally proven to correctly achieve the desired global behavior. The work of *Melo* and *Veloso* [53.145] takes a different approach to the synthesis of local decision policies by making use of the decentralized sparse-interaction Markov decision process. This technique allows agents to recognize states when interactions with other robots might occur, thus enabling them to choose better motions based on possible future inter-robot actions. *Tsiotras* and *Castro* [53.146] synthesize local controllers by generalizing the standard consensus algorithm, applied to geometric pattern formation.

A common theme in most of the works cited above is that they first make use of formal methods to describe the desired macro-level behavior, and then show how to use this macro-level goal to synthesize individual robot controllers. Another use of formal methods is to show how the individual goals of robot team members can be considered collectively, with the objective of maximizing the system's achievement of individual goals. Toward this end, game-theoretic techniques have been shown useful in a variety of distributed robot formulations. *Cheng* and *Dasgupta* [53.147] make use of the game-theoretic technique called Weighted Voting Games to address the problem of multi-robot team formation control amidst obstacles. *Taheri* et al. [53.148] also make use of game-theoretic principles, building upon the Local Interaction Game diffusion model to investigate how a small number of agents can influence the global society's behavior through local interactions.

An interesting question in the design of distributed robot coordination mechanisms is the extent to which identical controllers can lead to diversity, specialization, or changes in robot behavior. *Hsieh* et al. [53.149] study the emergence of specialization in robot swarms by making use of a distributed adaptation algorithm. They present a top-down analytical approach that defines the system equilibrium using waiting time parameters, and then present adaptive optimization strategies that converge to the optimal configurations that achieve system equilibrium. Temporal changes in system-level swarm behavior are addressed by *Hoff* et al. [53.150] who show how a swarm can change and improve its foraging behavior by switching between algorithms based on the environment in which the swarm finds itself.

Once the distributed controller is synthesized, most of the works mentioned above presume that individual robots execute their controller successfully. The typical presumption is that large swarms of interchangeable robots automatically result in robust and scalable swarm behavior. However, this presumption is challenged by *Winfield* and *Nembrini* [53.151], who illustrate that overall swarm reliability quickly falls in the presence of worst-case, partially failed robots. They conclude that future large scale swarm systems must develop new approaches for achieving high levels of fault tolerance. One example approach is shown in VIDEO 194.

53.6 Modular Robotics

The modular robotics field began with a paper presented at International Conference on Robotics and Automation in the Spring of 1988 by *Fukuda* and *Nakagawa* [53.152] describing the abstract concept of a reconfigurable robotic system that can assume different shapes and envisioned a robot system composed of different types of modules that can combine to accomplish a variety of tasks. Over the past twenty years, modular robotics research developed many facets: hardware design; planning and control algorithms; the trade-off between hardware and algorithmic complexity; efficient simulation; and system integration.

Modular robots are collections of physically connected, electromechanically active modules that, as a whole, form robotic systems that exhibit capabilities greater than those of the individual modules. Typically modular robots can change their shape or configuration in order to adapt to a variety of different tasks. For example, a collection of modules could reconfigure from a closed chain that rolls quickly over open ground to a legged robot that more easily traverses rough terrain. Modular robots are typically touted for their adaptability, their fault-tolerance, and the relative simplicity of the unit modules. Modular robotic systems can be described and classified on several axes using a variety of properties. In what follows, we choose the traditional route of classifying modular robotic systems by the geometry of the system: chain, lattice, truss, or free form. For a more detailed history of the modular robotics field, consult [53.153, 154].

53.6.1 Chain Systems

The defining characteristic of chain-type modular robot systems is the fact that the modules, when connected to their neighbors, are arranged in a chain. These chains may be one-dimensional, or two-dimensional, but three-dimensional chains are not as common. The fact that a chain-type modular robot is two-dimensional, or even one-dimensional, does not mean that it cannot operate in three dimensions. In fact, snake-like modular robots composed of segments with orthogonal joints are quite common.

One of the first chain-type modular robotic systems was the polypod system developed by *Yim* [53.155, 156] (VIDEO 196). The polypod system was composed of two types of modules: segments and nodes. It could form a variety of shapes including rolling loops and hexapods, and it went on to inspire many other chain-based systems. One was the CONRO system [53.157–159] in which each module was composed of two orthogonal servo motors controlling each module's pitch and yaw.

Murata et al. developed the M-TRAN modular robotic system [53.160–163] which has undergone multiple revisions and improvements. In [53.161], Kamimura et al. employ a set of interconnected, out of phase oscillators to achieve walking gaits in the M-TRAN system. *Marbach* and *Ijspeert* improved upon the ability of systems like M-TRAN to generate gaits in real-time by applying function optimization to their modular system, YaMoR [53.164]. *Murata* et al. added cameras to the M-TRAN system so that a set of M-TRAN modules could separate, perform independent tasks, and then rejoin into a larger structure [53.163].

The ATRON system [53.165, 166] was developed to improve upon the M-TRAN. Lund et al. wanted to keep M-TRAN's ability to form dense lattices while taking advantage of the two orthogonal degrees of freedom, (pitch and yaw), found in the CONRO system. The Superbot system [53.167] also builds upon on the mechanical design of M-TRAN by adding an additional degree of rotational freedom between the two existing rotation axes.

The PolyBot is chain-type modular robot [53.168, 169] with a single rotational degree of freedom. PolyBot evolved into CKBot which has demonstrated the ability to reassemble itself after being accidentally or intentionally destroyed [53.170]. The Molecube system [53.171], developed by *Lipson* et al. is another example of a chain-type modular system with only one degree of freedom but still able to achieve interesting three-dimensional (3-D) configurations. Lipson et al. showed that a short chain of Molecube modules, along with some free modules, can self-replicate.

Yim et al. designed another unique chain-type system named RATChET [53.172] which uses a connected chain of inter-latching right angle tetrahedrons to form structures. Neighboring RATChET modules latch together when the angle between them passes some critical value, and they unlatch through the use of shape memory alloy (SMA) springs when heated beyond 70 °C. Interestingly, the RATChET modules possess no intelligence. Instead, they rely on an intelligent external actuator which rotates to control one end of the dangling chain. One unique property of the RATChET system is its relatively strength.

53.6.2 Lattice Systems

Lattice-type modular robot systems are collections of interconnected robotic modules in which the units are situated at the intersection points of a two or three dimensional grid. (A 1-D (one-dimensional) lattice system is simply a chain-type robot.) The main characteris-

tic separating a lattice system from a densely configured chain-type robot is the density of the interconnections between the modules. In a lattice-type system, each module is typically connected to all of its neighbors. In a dense chain-type system, two modules may be neighbors, but they will not be physically connected.

Additionally, lattice-type systems tend to be built with modules that contain no rotational degrees of freedom. While the modules in a lattice system typically have mechanisms which enable the modules to move relative to, and bond with, their neighbors, they generally cannot bend themselves. In comparison, chain-type systems are often built from modules that contain one or more rotational degrees of free so that the modules can flex like links in a chain. There is some overlap between between the two types of system.

Chirikjian et al. developed one of the first lattice-based modular robotic systems [53.173–175] (VIDEO 198) in which the modules are deformable hexagons capable of bonding with their neighbors. Others, such as *Walter* et al. [53.176] further analyzed these hexagonal type systems to create distributed motion planners capable of reconfiguring the system from one state to another.

Murata et al. were also early contributors to the development of lattice-based modular robotic systems with their development of a roughly hexagonal module capable of rolling around its neighbors in two dimensions [53.177, 178]. *Kurokawa* et al. presented a three dimensional adaptation [53.179] composed of cubes with six protruding arms capable of rotation. *Yoshida* et al. improved on this system with a new design that used shape-memory alloy actuators to rotate one robot module around the perimeter of a neighbor [53.180].

One of the simplest lattice systems is the the Digital Clay project [53.181]. The system was a set of completely passive modules that relied on the user to make changes to it topology. The 2.5 cm rhombic dodecahedrons were able to sense and communicate with their neighbors in order to create a virtual model of the physical arrangement of modules.

Rus et al. also explored the idea of 3-D modules capable reconfiguration through a series of latchings, rotations, and unlatchings with the Molecule system [53.182–185]. In [53.186, 187], *Vona* and *Rus* describe a different type of deformable lattice system. The Crystal system is composed of square modules able to expand and contract by a factor of two in the x–y plane. *Suh* et al. expanded on the Crystal concept with the Telecubes [53.188] that could move in three dimensions by expanding all six faces.

Chiang and *Chirikjian* analyzed how to perform motion planning in a lattice of rigid cubic modules able to slide past each other [53.189]. The CHOBIE robot developed by *Koseki* [53.190] is able to actually perform the sliding motion assumed by *Chiang* and *Chirikjian* in [53.189]. More recently, *An* developed the EM-Cube system [53.191] which is also capable of sliding motion.

Another unique lattice is the I-Cube developed by *Khosla* et al. [53.192, 193]. The 3-D I-Cube system consists of passive cubes which are connected by active links with three rotational degrees of freedom that are able to grab, reposition, and release the cubes. The 3-D I-Cube system was an improvement of the two-dimensional (2-D) system [53.194] developed by *Hosokawa* et al. for rearranging cubic modules in a vertical plane.

Goldstein et al. initiated the Claytronics project by publishing several papers [53.195, 196] proposing lattice-based *claytronic atoms* or catoms. These vertically-oriented cylindrical robots, which were incapable of independent motion, used 24 electromagnets around their perimeters to achieve rolling locomotion about their neighbors. Goldstein et al. envisioned a system in which millions of smaller catoms could form arbitrary shapes using a randomized algorithm that avoided conveying a complete description of the shape to each module in the system.

The catoms continue to evolve. One of the newest instantiations [53.197] employs hollow cylinders rolled from SiO_2 rectangles patterned with aluminum electrodes. The authors hope that two of these cylinders, when placed in close proximity with their axes aligned, will be able to rotate with respect to one another using electrostatic forces. Specifically, the electrodes, (which reside on the inside of each cylinder and are electrically isolated by the SiO_2), will be charged so that they attract and repel mirror charges on the neighboring cylinder in a way that causes rotation. Currently, the system appears to be constrained to form 2-D structures. The authors claim the completed system will have a yield strength similar to that of plastic and that the modules will be able to transfer power and communication signals capacitively from neighbor to neighbor.

The Claytronics project has proposed, but not yet demonstrated with hardware, the use of sub-millimeter intelligent particles as sensing and replication devices [53.198]. In particular, Pillai et al. present a theoretical 3-D fax machine in which the object to be *faxed* is immersed in a container of intelligent particles that sense and encode the object's dimensions. At the receiving end, these same Claytronic particles decode the shape description sent by the transmitter and bond together to replicate the original object. Unlike our approach, Pillai's approach is completely centralized and relies on an external computer for computation.

White et al. developed hardware and algorithms for several 2-D stochastically-driven self-assembling sys-

tems [53.199]. To form specific shapes, each module is provided with a representation of the desired shape and decides, based on its location in the structure, whether to allow other modules to bond to its faces. *Lipson* et al. extended their 2-D system to 3-D [53.200–202] by using cubic modules suspended in turbulent fluid to achieve self-assembly and reconfiguration. As the free modules circulate in the fluid, they pass by a growing structure of assembled modules. When they come close enough, they are accreted onto the structure. The modules attract or repel each other with fluid suction or positive pressure. Early versions of the system used modules with interval values that could redirect these suction forces. More recently, *Lipson*'s group has worked to move the intelligence and actuation capabilities from the modules to the tank in which the modules circulate [53.203].

The Miche system [53.204] consisting of 45 mm cubic modules capable of mating with their neighbors using mechanically switchable permanent magnets. Each module contains three switchable magnets, each of which mated with a steel face on a neighboring module. Because the connectors were gendered, any collection of modules had to be assembled by hand so that the connectors were always oriented correctly, but the system was capable of self-disassembling to form 3-D structures. The Robot Pebbles (VIDEO 211) are based, at least in principle, on the Miche modules.

One of the newest lattice-type modular robotics is an aerial system composed of identical, hexagonal, single-rotor modules [53.205]. A group of modules may connect to form a flying platform with an arbitrary arrangement of multiple rotors. In addition to the ability to fly, each module contains wheels so that the system may self-reconfigure on the ground for the specific task at hand.

53.6.3 Truss Systems

Truss systems, as their name implies, are modular robotic systems in which the modules are nodes and edges in a truss structure. Both the trusses and connectors may be active in such systems. Unlike the lattice-based systems, truss-based systems do not need to operate on any regular lattice. Most truss-based systems under development employ struts that expand or contract to achieve structural deformation. One of the first such system to do so was Tetrobot [53.206]. The Odin system, conceived by *Lyder* et al. [53.207, 208] consists of three physically different types of modules: active strut modules capable of changing their length; passive strut modules of fixed length; and joint modules. The biologically inspired Morpho system [53.209] developed by *Nagpal* et al. is similar to Odin. It also uses active links, passive links, and connector cubes.

53.6.4 Free-Form Systems

Free-form systems are able to aggregate modules in at least semi-arbitrary positions. One such system is the Slimebot [53.210, 211]. The system consists of identical vertical cylindrical modules that move on a horizontal plane. The perimeter of each module is covered by six gender-less hook and loop patches used to bond with neighboring modules. These patches oscillate radially in and out from the center of the body. By controlling the frequency and phase of the oscillations between neighbors, the system can achieve aggregate motion in a given direction.

Researchers are also developing algorithms for free-form systems. *Funiak* et al. developed a localization algorithm [53.212] that is capable of localizing tens-of-thousands of irregularly packed modules in 3-D. Rubenstein and Shen developed a number of shape formation algorithms for collections of two-dimensional modules. These algorithms allow an arbitrary-sized collection of modules to form arbitrary scale-independent shapes [53.213, 214]. Once the shape is formed, modules can be added to or removed from the system, and the system will reconfigure itself to incorporate the new modules. The resulting shape will grow or shrink, but its basic form will remain unchanged. Recently, *Rubenstein* et al. developed a 1000 modules hardware platform on which to deploy these algorithms [53.215].

Researchers have also explored the use of folding to create reconfigurable foldable systems [53.216, 217]. These systems use flexible wiring and shape memory alloy actuators embedded in composite sheets to programmatically create origami-inspired shapes. By controlling which actuators are energized, the system can form multiple different shapes.

53.6.5 Self-Assembling Systems

Self-assembling modular robotic systems are collection of modules that are capable of autonomously coalescing and bonding with their neighbors to form a greater structure. The result is often robotic, but it need not be. Whether a system is capable of self-assembling is independent of whether it is free-form, a chain, a lattice, or a truss-based system. Almost all of aforementioned modular robot systems rely on human intervention to assemble. In an attempt to automate the process of creating intricate modular robotic systems, researchers have attempted to mimic and improve upon natural self-assembling systems. *Whitesides* et al. investigated a wide variety of engineered self-assembling systems [53.218–220].

Miyashita et al. performed a more theoretical analysis of self-assembly using pie-shaped pieces

to form complete circles [53.221] from pie-shaped pieces. In the process, they followed *Hosokawa* et al.'s lead [53.222] and modeled the system as a chemical reaction. *Shimizu* and *Suzuki* developed a system of passive modules capable of self-repair when placed on a vibrating table [53.223].

Computer scientists have also investigated theoretical aspects of self-assembly in the context of 2-D tiles which selectively bond with their neighbors to form simple well-defined shapes like squares [53.224–226]. Each side of every tile in the system has an associated bonding strength. When two tiles collide, they remain attached only if their cumulative bond strength exceeds a globally defined system entropy. To form a specific shape, one must design a set of tiles with the appropriate bonding strengths.

Klavins et al. worked to develop intelligent self-assembling systems that employ triangular modules driven by oscillating fans on an air table to self-assemble different shapes [53.227]. The authors employ knowledge of the module's local topology and internal module state so that each module decides, in a distributed fashion, when to maintain or break a connection with its immediate neighbors. *Griffith* et al. also worked with intelligent modules capable of selective bonding to show that self-assembling systems may self-replicate [53.228].

Rus et al. [53.185] present the first generic rule-based approach to self-assembly, shape formation, and locomotion by reconfiguration. The rules can be used on any modular robot system that can implement the sliding cube model of relative motion. The result is an abstract set of rules for each of these tasks, that can be compiled down to module motions, taking into account how the physical module implementes translation and convex and concave transitions.

Jones and *Matarić* [53.229] presented rule-based approach to self-assembly termed transition rule sets. In particular, they present a method that, given a goal structure, produces a set of rules shared among all modules that govern when and where new modules are allowed to attach to the growing structure. *Kelly* and *Zhang* [53.230] expanded on this work by optimizing the size of the rule sets used to form a specific shape. *Werfel* [53.231] also applied the idea of a transition rule set when studying the use of swarms to assemble complex structures from passive materials.

Other groups have attempted to make self-assembly more deterministic. The MEMS (microelectromechanical system) robots developed by *Donald* et al. [53.232, 233] consists of thin ($7-20\,\mu m$), rectangular ($\approx 260\,\mu m \times 60\,\mu m$), scratch-drive devices capable of moving on an insulating substrate embedded with electrodes. The authors used four of these robots to build larger composite structures. The Sitti group has developed a similar system of micro-meter sized robots [53.234]. Instead of using a scratch drive for locomotion, the robots are manipulated by external magnetic fields. The authors can electrostatically clamp any number of robots to the stage on which they move. With all but one robot immobilized, the remaining robot may be moved independently. The system naturally self-assembles because the robots contain permanent magnets that attract their neighbors.

The majority of existing self-assembly systems aim to form structures in one of two ways. Some systems such as [53.221, 223–226] use a collection of application specific differentiated modules, that are only capable of assembling in a particular fashion to form a specific shape. In contrast, other systems such as [53.199–201, 227, 229–231, 235] use completely generic modules with more computation and communication ability embedded in each module. Both types of systems aim to form complex shapes in a direct manner: as these structures grow from a single module, new modules are only allowed to attach to the structure in specific locations.

An alternative approach eliminates many of the complexities of shape formation by active assembly by using dissasemly for shape formation. The Smart Pebble system [53.154, 204, 236, 237] employs a set of distributed algorithms to perform two discrete steps: 1) rely on stochastic forces to self-assemble a close-packed crystalline lattice of modules and 2) use the process of self-disassembly to remove the extra material from this block leaving behind the goal structure. By approaching shape formation in this manner, the entire shape-formation process is sped-up and the robustness of the system is increased.

53.7 Heterogeneity

Robot heterogeneity can be defined in terms of variety in robot behavior, morphology, performance quality, size, and cognition. In most large-scale multirobot systems work, the benefits of parallelism, redundancy, and solutions distributed in space and time are obtained through the use of homogeneous robots, which are completely interchangeable (i. e., the swarm approach, as described in Sect. 53.4). However, certain complex applications of large-scale robot teams may require the simultaneous use of multiple types of sensors and

robots, all of which cannot be designed into a single type of robot. Some robots may need to be scaled to smaller sizes, which will limit their payloads, or certain required sensors may be too expensive to duplicate across all robots on the team. Other robots may need to be large to carry application-specific payload or sensors, or to navigate long distances in a limited time. These applications, therefore, require the collaboration of large numbers of heterogeneous robots.

The motivation for developing heterogeneity in multirobot teams is thus twofold: heterogeneity may be a design feature beneficial to particular applications, or heterogeneity may be a necessity. As a design feature, heterogeneity can offer economic benefits, since it can be easier to distribute varying capabilities across multiple team members rather than to build many copies of monolithic robots. Heterogeneity can also offer engineering benefits, as it may simply be too difficult to design individual robots that incorporate all of the sensing, computational, and effector requirements of a given application. Heterogeneity in behavior may also arise in an emergent manner in physically homogeneous teams, as a result of behavior specialization.

A second compelling reason to study heterogeneity is that it may be a necessity, in that it is nearly impossible in practice to build a truly homogeneous robot team. The realities of individual robot design, construction, and experience will inevitably cause a multirobot system to drift to heterogeneity over time. This is recognized by experienced roboticists, who have seen that several copies of the same model of robot can vary widely in capabilities due to differences in sensor tuning, calibration, etc. Over time, even minor initial differences among robots will grow due to individual robot drift and wear and tear. The implication is that, to employ robot teams effectively, we must understand diversity, predict how it will impact performance, and enable robots to adapt to the diverse capabilities of their peers. In fact, it is often advantageous to build diversity explicitly into the design of a robot team.

There are a variety of research challenges in heterogeneous multirobot systems. A particular challenge to achieving efficient autonomous control is when overlap in team member capabilities occurs, thus affecting task allocation or role assignments [53.238]. Techniques as described in Sect. 53.6 can typically deal with heterogeneous robots for the purposes of task allocation. Another important topic in heterogeneity is how to recognize and quantify heterogeneity in multirobot teams. Some types of heterogeneity can be evaluated quantitatively, using metrics such as the social entropy metric developed by *Balch* [53.239]. Most research in heterogeneous multirobot systems assumes that robots have a common language and a common understanding of symbols in their language; developing a common understanding of communicated symbols among robots with different physical capabilities is a fundamental challenge, addressed by *Jung* and *Zelinsky* [53.240] (VIDEO 200).

As discussed in Sect. 53.2, one of the earliest research demonstrations of heterogeneity in physical robot teams was in the development of the ALLIANCE architecture by *Parker* [53.9]. This work demonstrated the ability of robots to compensate for heterogeneity in team members during task allocation and execution. *Murphy* has studied heterogeneity in the context of marsupial robot deployment, where a *mothership* robot assists smaller robots in applications such as search and rescue [53.241] (VIDEO 206). *Grabowski* et al. [53.134] developed modular millibots for surveillance and reconnaissance that could be composed of interchangeable sensor and effector components, thus creating a variety of different heterogeneous teams. *Simmons* et al. [53.11] demonstrated the use of heterogeneous robots for autonomous assembly and construction tasks relevant to space applications. *Sukhatme* et al. [53.242] demonstrated a helicopter robot cooperating with two ground robots in tasks involving marsupial-inspired payload deployment and recovery, cooperative localization, and reconnaissance and surveillance tasks, as shown in Fig. 53.14. *Parker* et al. [53.243] demonstrated assistive navigation for sensor network deployment using a more intelligent leader robot for guiding navigationally challenged simple sensor robots to goal locations, as part of a larger demonstration by *Howard* et al. [53.244] of 100 robots performing exploration, mapping, deployment, and detection. *Chaimowicz* et al. [53.245] demonstrated a team of aerial and ground robots cooperating for surveillance applications in urban environments. *Parker* and *Tang* [53.246] developed ASyMTRe (automated



Fig. 53.14 Heterogeneous team of an air and two ground vehicles that can perform cooperative reconnaissance and surveillance

synthesis of multirobot task solutions through software reconfiguration), which enables heterogeneous robots to share sensory resources to enable the team to accomplish tasks that would be impossible without tightly coupled sensor sharing.

Many open research issues remain to be solved in heterogeneous multirobot teams; for example, the issue of optimal team design is a very challenging problem. Clearly, the required behavioral performance in a given application dictates certain constraints on the physical design of the robot team members. However, it is also clear that multiple choices may be made in designing a solution to a given application, based upon cost, robot availability, ease of software design, flexibility in robot use, and so forth. Designing an optimal robot team for a given application requires significant analysis and consideration of the tradeoffs in alternative strategies.

53.8 Task Allocation

In many multirobot applications, the mission of the team is defined as a set of tasks that must be completed. Each task can usually be worked on by a variety of different robots; conversely, each robot can usually work on a variety of different tasks. In many applications, a task is decomposed into independent subtasks [53.9], hierarchical task trees [53.247], or roles [53.11, 245, 248, 249] either by a general autonomous planner or by the human designer. Independent subtasks or roles can be achieved concurrently, while subtasks in task trees are achieved according to their interdependence. Once the set of tasks or subtasks have been identified, the challenge is to determine the preferred mapping of robots to tasks (or subtasks). This is the *task allocation* problem.

The details of the task allocation problem can vary in many dimensions, such as the number of robots required per task, the number of tasks a robot can work on at a time, the coordination dependencies among tasks, and the time frame for which task assignments are determined. *Gerkey* and *Matarić* [53.250] defined a taxonomy for task allocation that provides a way of distinguishing task allocation problems along these dimensions, which is referred to as the multirobot task allocation (MRTA) taxonomy.

53.8.1 Taxonomy for Task Allocation

Generally, tasks are considered to be of two principal types: *single-robot tasks* (SR, according to the MRTA taxonomy) are those that require only one robot at a time, while *multirobot tasks* (MR) are those that require more than one robot working on the same task at the same time. Commonly, single-robot tasks that have minimal task interdependencies are referred to as *loosely coupled tasks*, representing a *weakly cooperative* solution. On the other hand, multirobot tasks are often considered to be sets of subtasks that have strong interdependencies. These tasks are therefore often referred to as *tightly coupled tasks* that require a *strongly cooperative* solution. The subtasks of a loosely coupled multirobot task require a high level of synchronization or coordination between subtasks, meaning that each task must be aware of the current state of the coordinated subtasks within a small time delay. As this time delay becomes progressively larger, coordinated subtasks become more loosely coupled, representing weakly cooperative solutions.

Robots can also be categorized as either *single-task robots* (ST), which work on only one task at a time or *multitask robots* (MT), which are able to make progress on more than one task at a time. Most commonly, task allocation problems assume robots are single-task robots, since more capable robots that perform multiple tasks in parallel are still beyond the current state of the art.

Tasks can either be assigned to optimize the instantaneous allocation of tasks (IA), or to optimize the assignments into the future (TA, for time-extended assignment). In the case of instantaneous assignment, no consideration is made for the effect of the current assignment on future assignments. Time-extended assignments attempt to assign tasks so that the performance of the team is optimized for the entire set of tasks that may be required, not just the current set of tasks that need to be achieved at the current time step.

Using the MRTA taxonomy, triples of these abbreviations are used to categorize various task allocation approaches, such as SR-ST-IA, which refers to an assignment problem in which single-robot tasks are assigned once to single-task robots. Different variations of the task allocation problem have different computational complexities. The easiest variant is the ST-SR-IA problem, which can be solved in polynomial time since it is an instance of the optimal assignment problem [53.251]. Other variants are much more difficult, and do not have known polynomial time solutions. For example, the ST-MR-IA variant can be shown to be an instance of the set partitioning problem [53.252], which is strongly *NP*-hard. The ST-MR-TA, MT-SR-IA, and MT-SR-TA variants have also all been shown to be *NP*-hard problems. Because these problems are computationally complex, most approaches to task allocation in multirobot teams generate approximate solutions.

53.8.2 Representative Approaches

Approaches to task allocation in multirobot teams can be roughly divided into behavior-based approaches and market-based (sometimes called negotiation-style or auction-based) approaches. The following subsections describe some representative architectures for each of these general approaches. Refer to [53.250] for a comparative analysis of some of these approaches, in terms of computation and communications requirements and solution quality.

Behavior-Based Task Allocation

Behavior-based approaches typically enable robots to determine task assignments without explicitly discussing individual tasks. In these approaches, robots use knowledge of the current state of the robot team mission, robot team member capabilities, and robot actions to decide, in a distributed fashion, which robot should perform which task.

One of the earliest architectures for multirobot task allocation that was demonstrated on physical robots was the behavior-based ALLIANCE architecture [53.9] and the related L-ALLIANCE architecture [53.10]. ALLIANCE addresses the ST-SR-IA and ST-SR-TA variants of the task allocation problem without explicit communication among robots about tasks. As described in Sect. 53.2.2, ALLIANCE achieves adaptive action selection through the use of motivational behaviors, which are levels of impatience and acquiescence within each robot that determine its own and its teammates' relative fitness for performing certain tasks. These motivations are calculated based upon the mission requirements, the activities and capabilities of teammates, and the robots' internal states. These motivations effectively calculate utility measures for each robot–task pair.

Another behavior-based approach to multirobot task allocation is broadcast of local eligibility (BLE) [53.253], which addresses the ST-SR-IA variant of task allocation. BLE uses a subsumption style behavior control architecture [53.254] that allows robots to efficiently execute tasks by continuously broadcasting locally computed eligibilities and only selecting the robot with the best eligibility to perform the task. In this case, task allocation is achieved through behavior inhibition. BLE uses an assignment algorithm that is very similar to *Botelho* and *Alami*'s M+ architecture [53.255].

Market-Based Task Allocation

Market-based (or negotiation-based) approaches typically involve explicit communications between robots about the required tasks, in which robots bid for tasks based on their capabilities and availability. The negotiation process is based on market theory, in which the team seeks to optimize an objective function based upon individual robot utilities for performing particular tasks. The approaches typically greedily assign subtasks to the robot that can perform the task with the highest utility.

Smith's contract net protocol (CNP) [53.256] was the first to address the problem of how agents can negotiate to collectively solve a set of tasks. The use of a market-based approach specifically for multirobot task allocation was first developed by *Botelho* and *Alami* with their M+ architecture [53.255]. In the M+ approach, robots plan their own individual plans for the task they have been assigned. They then negotiate with other teammates to incrementally adapt their actions to suit the team as a whole, through the use of social rules that facilitate the merging of plans.

Since these early developments, many alternative approaches to market-based task allocation have been developed. A thorough survey on the current state of the art in market-based techniques for multirobot task allocation is given in [53.257], comparing alternative approaches in terms of solution quality, scalability, dynamic events and environments, and heterogeneous teams.

Most of the current approaches in market-based task allocation address the ST-SR problem variant, with some approaches (e.g., [53.11, 258–260]) dealing with instantaneous assignment (IA), and others (e.g., [53.135, 261–263]) addressing time-extended assignments (TA). More recent methods are beginning to address the coaltion formation problem, which is the allocation of multirobot tasks (i. e., the MR-ST problem variant), including [53.246, 264–269]. An example approach to the MR-MT problem variant is found in [53.270].

Some representative market-based techniques include MURDOCH [53.258], TraderBots [53.247, 263], and Hoplites [53.265]. The MURDOCH approach [53.258] employs a resource-centric, publish–subscribe communication model to carry out auctions, which has the advantage of anonymous communication. In this approach a task is represented by the required resources, such as the environmental sensors. The methods for how to use such a sensor to generate satisfactory results is preprogrammed into the robot.

The TraderBots approach [53.247, 263] applies market economy techniques for generating efficient and robust multirobot coordination in dynamic environments. In a market economy, robots act based on selfish interests. A robot receives revenue and incurs cost when trying to accomplish a task. The goal is for robots to

trade tasks through auctions/negotiations such that the team profit (revenue minus cost) is optimized.

The Hoplites approach [53.265] focuses on the selection of an appropriate joint plan for the team to execute by incorporating joint revenue and cost into the bid. This approach couples planning with passive and active coordination strategies, enabling robots to change coordination strategies as the needs of the task change. Strategies are predefined for a robot to accomplish a selected plan.

Some alternative approaches formulate the objects to be assigned as *roles*, which typically package a set of tasks and/or behaviors that a robot should undertake when acting in a particular role. Roles can then be dynamically assigned to robots in a similar manner as in the auction-based approaches [53.11, 248].

53.9 Learning

Multirobot learning is the problem of learning new cooperative behaviors, or learning in the presence of other robots. The other robots in the environment, however, have their own goals and may be learning in parallel [53.271]. The challenge is that having other robots in the environment violates the Markov property that is a fundamental assumption of single-robot learning approaches [53.271]. The multirobot learning problem is particularly challenging because it combines the difficulties of single-robot learning with multiagent learning. Particular difficulties that must be considered in multirobot learning include continuous state and action spaces, exponential state spaces, distributed credit assignment, limited training time and insufficient training data, uncertainty in sensing and shared information, nondeterministic actions, difficulty in defining appropriate abstractions for learned information, and difficulty of merging information learned from different robot experiences.

The types of applications that have been studied for multirobot learning include multitarget observation [53.272, 273], air fleet control [53.274], predator–prey [53.137, 275, 276], box pushing [53.277], foraging [53.23], and multirobot soccer [53.140, 278]. Particularly challenging domains for multirobot learning are those tasks that are *inherently* cooperative. Inherently cooperative tasks are those that cannot be decomposed into independent subtasks to be solved by individual robots. Instead, the utility of the action of one robot is dependent upon the current actions of the other team members. This type of task is a particular challenge in multirobot learning, due to the difficulty of assigning credit for the individual actions of the robot team members.

The credit assignment problem is a particular challenge, since it is difficult for a robot to determine whether the fitness (either good or bad) is due to its own actions, or due to the actions of another robot. As discussed by *Pugh* and *Martinoli* in [53.279], this problem can be especially difficult in situations where robots do not explicitly share their intentions. Two different variations of the credit assignment problem are common in multirobot learning. The first is when robots are learning individual behaviors in the presence of other robots that can affect their performance. The second is when robots are attempting to learn a task with a shared fitness function. It can be difficult to determine how to decompose the fitness function to appropriately reward or penalize the contributions of individual robots.

While learning has been explored extensively in the area of single-robot systems (see, for example, the discussion of learning in behavior-based systems in Chap. 13, and a discussion of fundamental learning techniques in Chap. 15) and in multi*agent* systems [53.280], much less work has been done in the area of multi*robot* learning, although the topic is gaining increased interest. Much of the work to date has focused on reinforcement learning approaches. Some examples of this multirobot learning research include the work by *Asada* et al. [53.281], who propose a method for learning new behaviors by coordinating previously learned behaviors using Q-learning, and apply it to soccer-playing robots. *Matarić* [53.8] introduces a method for combining basic behaviors into higher-level behaviors through the use of unsupervised reinforcement learning, heterogeneous reward functions, and progress estimators. This mechanism was applied to a team of robots learning to perform a foraging task. *Kubo* and *Kakazu* [53.282] proposed another reinforcement learning mechanism that uses a progress value for determining reinforcement, and applied it to simulated ant colonies competing for food. *Fernandez* and *Parker* [53.272] apply a reinforcement learning algorithm that combines supervised function approximation with generalization methods based on state-space discretization, and apply it to robots learning the multi-object tracking problem. *Bowling* and *Veloso* [53.271] developed a general-purpose, scalable learning algorithm called GraWoLF (gradient-based win or learn fast), which combines gradient-based policy learning

techniques with a variable learning rate, and demonstrated the results in the adversarial multirobot soccer application.

Other multirobot learning approaches not based on reinforcement include *Parker*'s L-ALLIANCE architecture [53.10], which uses parameter tuning, based on statistical experience data, to learn the fitness of different heterogeneous robots in performing a set of tasks. *Pugh* and *Martinoli* [53.279] apply particle swarm optimization techniques to distributed unsupervised robot learning in groups, for the task of learning obstacle avoidance.

53.10 Applications

Many real-world applications can potentially benefit from the use of multiple mobile robot systems. Example applications include container management in ports [53.283], extraplanetary exploration [53.284], search and rescue [53.241], mineral mining, transportation, industrial and household maintenance, construction [53.11], hazardous waste cleanup [53.9], security [53.285, 286], agriculture, and warehouse management [53.287] (VIDEO 210). Multiple robot systems are also used in the domain of localization, mapping, and exploration; Chap. 46 mentions some of the work in multirobot systems applied to these problems. Parts F and G of this handbook outline many application areas that are relevant not only to single-robot systems, but also to multiple mobile robot systems. To date, relatively few real-world implementations of these multirobot systems have occurred, primarily due to the complexities of multiple robot systems and the relative newness of the supporting technologies. Nevertheless, many proof-of-principle demonstrations of physical multirobot systems have been achieved, and the expectation is that these systems will find their way into practical implementations as the technology continues to mature.

Research in multiple mobile robot systems is often explored in the context of common application test domains. While not yet elevated to the level of benchmark tasks, these common domains do provide opportunities for researchers to compare and contrast alternative strategies to multirobot control. Additionally, even though these common test domains are usually just laboratory experiments, they do have relevance to real-world applications. This section outlines these common application domains; see also [53.2] and [53.288] for a discussion of these domains and a more detailed listing of related research.

53.10.1 Foraging and Coverage

Foraging is a popular testing application for multirobot systems, particularly for those approaches that address swarm robotics, involving very large numbers of mobile robots. In the foraging domain, objects such as pucks or simulated food pellets are distributed across the planar terrain, and robots are tasked with collecting the objects and delivering them to one or more gathering locations, such as a home base. Foraging lends itself to the study of weakly cooperative robot systems, in that the actions of individual robots do not have to be tightly synchronized with each other. This task has traditionally been of interest in multirobot systems because of its close analogy to the biological systems that motivate swarm robotics research. However, it also has relevance to several real-world applications, such as toxic waste cleanup, search and rescue, and demining. Additionally, since foraging usually requires robots to completely explore their terrain in order to discover the objects of interest, the *coverage* domain has similar issues to the foraging application. In coverage, robots are required to visit all areas of their environment, perhaps searching for objects (such as landmines) or executing some action in all parts of the environment (e.g., for floor cleaning). The coverage application also has real-world relevance to tasks such as demining, lawn care, environmental mapping, and agriculture.

In foraging and coverage applications, a fundamental question is how to enable the robots to explore their environments quickly without duplicating actions or interfering with each other. Alternative strategies can include basic stigmergy [53.14], forming chains [53.120], and making use of heterogeneous robots [53.131]. Other research demonstrated in the foraging and/or coverage domain includes [53.23, 132, 289–294].

53.10.2 Flocking and Formations

Coordinating the motions of robots relative to each other has been a topic of interest in multiple mobile robot systems since the inception of the field. In particular, much attention has been paid to the flocking and formation control problems (VIDEO 217, VIDEO 293). The flocking problem could be viewed as a subcase of the formation control problem, requiring robots to move together along some path in the aggregate, but with only minimal requirements for paths taken by specific robots. Formations are stricter, requiring robots to

maintain certain relative positions as they move through the environment. In these problems, robots are assumed to have only minimal sensing, computation, effector, and communications capabilities. A key question in both flocking and formation control research is determining the design of local control laws for each robot that generate the desired emergent collective behavior. Other issues include how robots cooperatively localize themselves to achieve formation control [53.133, 295], and how paths can be planned for permutation-invariant multirobot formations [53.296].

Early solutions to the flocking problem in artificial agents were generated by *Reynolds* [53.297] using a rule-based approach. Similar behavior- or rule-based approaches have been used physical robot demonstrations and studies, such as in [53.121, 298]. These earlier solutions were based on human-generated local control rules that were demonstrated to work in practice. More recent work is based on control theoretic principles, with a focus on proving stability and convergence properties in multirobot team behaviors. Examples of this work include [53.128, 299–307]. Refer to [53.308, 309] for surveys of relevant control theoretic work.

53.10.3 Object Transportation and Cooperative Manipulation

Some of the earliest work in swarm robotics was aimed at the object transportation task [53.13, 123, 310–313], which requires a team of robots to move an object from its current position in the environment to some goal destination (VIDEO 193). The primary benefit of using collective robots for this task is that the individual robots can combine forces to move objects that are too heavy for individual robots working alone or in small teams. However, the task is not without its challenges; it is non-trivial to design decentralized robot control algorithms that can effectively coordinate robot team members during object transportation. A further complication is that the interaction dynamics of the robots with the object can be sensitive to certain object geometries [53.314, 315] and object rotations during transportation [53.315], thus exacerbating the control problem.

Object transportation and cooperative manipulation are popular domains for demonstrating multirobot cooperation, because they offer a clear domain where close coordination and cooperation is required. A common type of object transportation – box pushing – requires robot teams to move boxes from their starting positions to defined goal configurations, sometimes along specified paths. Typically, box pushing operates in the plane, and the assumption is made that the boxes are too heavy or too long to enable single robots to push alone. Sometimes there are several boxes to be moved, with ordering dependencies constraining the sequence of motions. Cooperative manipulation is similar, except it requires robots to lift and carry objects to a destination. This test bed domain lends itself to the study of strongly cooperative multirobot strategies, since robots often have to synchronize their actions to successfully execute these tasks. The domain of box pushing and cooperative manipulation is also popular because it has relevance to several real-world applications [53.288], including warehouse stocking, truck loading and unloading, transporting large objects in industrial environments, and assembly of large-scale structures.

Researchers usually emphasize different aspects of their cooperative control approach in the box pushing and cooperative manipulation domain. For example, *Kube* and *Zhang* [53.13] (VIDEO 199) demonstrate how swarm-type cooperative control techniques could achieve box pushing (Fig. 53.15), *Parker* [53.10, 316] illustrates aspects of adaptive task allocation and learning, *Donald* et al. [53.317] (VIDEO 208) illustrates concepts of information invariance and the interchangeability of sensing, communication, and control, and *Simmons* et al. [53.11] demonstrate the feasibility of cooperative control for building planetary habitats.

In general, the manipulation techniques used for collective object transportation can be grouped into three primary methods [53.318]: pushing, grasping, and caging. The pushing approach [53.10, 11, 13, 316, 317] requires contact between each robot and the object, in order to impart force in the goal direction; however, the robots are not physically connected with the object. In the grasping approach [53.123, 142, 310–312, 319–322], each robot in the team physically attaches to the object being transported. See for example Fig. 53.16

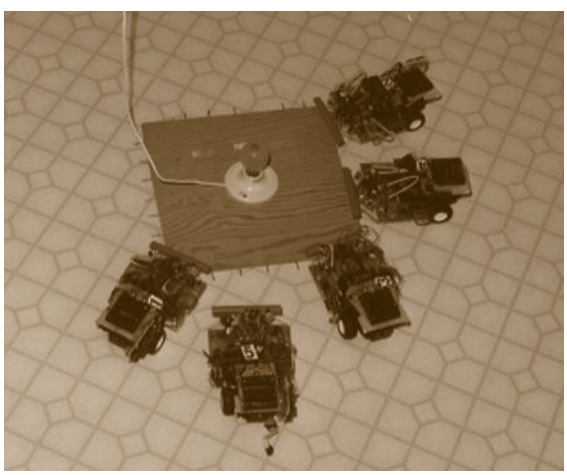


Fig. 53.15 Collective pushing of lighted box

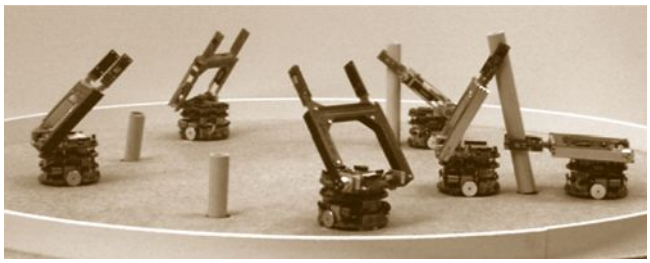


Fig. 53.16 Cooperative stick pulling

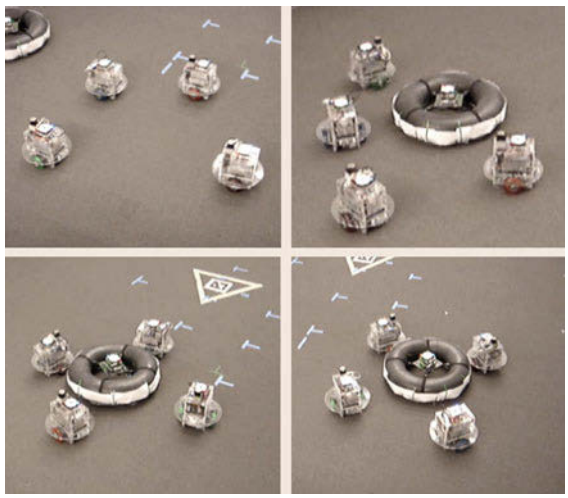


Fig. 53.17 Collective transport via caging

for cooperative stick pulling work of [53.320]. Finally, the caging approach [53.323–326] (VIDEO 292) involves robots encircling the object so that the object moves in the desired direction, even without the constant contact of all the robots with the object. See Fig. 53.17 for an example of collective transport via caging, from [53.323].

A closely related task is that of collective construction and wall building. The objective of the collective construction and wall building task is for robots to build structures of a specified form, in either 2-D or 3-D. This task is distinguished from self-reconfigurable robots, whose bodies themselves serve as the dynamic structure. *Werfel* and *Nagpal* have extensively explored this topic (VIDEO 216), developing distributed algorithms that enable simplified robots to build structures based on provided blueprints, both in 2-D [53.327–329] and in 3-D [53.330]. In their 3-D approach, the system consists of idealized mobile robots that perform the construction, and smart blocks that serve as the passive structure. The robots' job is to provide the mobility, while the blocks' role is to identify places in the growing structure at which an additional block can be placed that is on the path toward obtaining the desired final structure. The goal of their work is to be able to deploy some number of robots and free blocks into a construction zone, along with a single block that serves as a seed for the structure, and then have the construction to proceed autonomously according to the provided blueprint of the desired structure.

Hardware challenges of collective robot construction are addressed by *Terada* and *Murata* [53.331]. In this work, a hardware design is proposed that defines passive building blocks, along with an assembler robot that constructs structures with the robots. Other related work on the topic of collective construction includes the work of *Wawerla* et al. [53.332], in which robots use a behavior-based approach to build a linear wall using blocks equipped with either positive or negative Velcro, distinguished by block color. Their results show that adding 1 bit of state information to communicate the color of the last attached block provides a significant improvement in the collective performance. The work by *Stewart* and *Russell* [53.333, 334] proposes a distributed approach to building a loose wall structure with a robot swarm.

Another type of construction is called blind bulldozing, which is inspired by a behavior observed in certain ant colonies. Rather than constructing by accumulating materials, this approach achieves construction by removing materials. This task has practical application in site clearing, such as would be needed for planetary exploration [53.335]. Early ideas of this concept were discussed by *Brooks* et al. in [53.336], which argues for large numbers of small robots to be delivered to the lunar surface for site preparation. *Parker* et al. [53.337], further develop this idea by proposing robots using force sensors to clear an area by pushing material to the edges of the work site.

A significant body of additional research has been illustrated in this domain; representative examples include [53.3, 6, 123, 258, 284, 310, 323, 338–344].

53.10.4 Multitarget Observation

The domain of multitarget observation requires multiple robots to monitor and/or observe multiple targets moving through the environment. The objective is to maximize the amount of time, or the likelihood, that the targets remain in view by some team member throughout task execution. The task can be especially challenging if there are more targets than robots. This application domain can be useful for studying strongly cooperative task solutions, since robots have to coordinate their motions or the switching of targets to follow in order to maximize their objective. In the context of multiple mobile robot applications, the planar version of this test bed was first introduced

in [53.345] as cooperative multirobot observation of multiple moving targets (CMOMMT). Similar problems have been studied by several researchers, and extended to more complex problems such as environments with complex topography or three-dimensional versions for multiple aerial vehicle applications. This domain is also related to problems in other areas, such as art gallery algorithms, pursuit evasion, and sensor coverage. This domain has practical application in many security, surveillance, and reconnaissance problems. Research applied to the multitarget observation problem in multirobot systems includes [53.138, 253, 346–353].

53.10.5 Traffic Control and Multirobot Path Planning

When multiple robots are operating in a shared environment, they must coordinate their actions to prevent interference. These problems typically arise when the space in which robots operate contains bottlenecks, such as networks of roadways, or when the robots take up a relatively large portion of the navigable space. In these problems, the open space can be viewed as a resource that robots must share as efficiently as possible, avoiding collisions and deadlocks. In this domain, robots usually have their own individual goals, and must work with other robots to ensure that they receive use of the shared space to the extent needed to achieve their goals. In some variants, the entire paths of multiple robots need to be coordinated with each other; in other variants, robots must simply avoid interfering with each other.

A variety of techniques have been introduced to address this problem, including traffic rules, subdividing the environment into single-ownership sections, and geometric path planning ([53.354] for an overview). Many of the earliest research approaches to this problem were based on heuristic approaches, such as predefining motion control (or traffic) rules that were shown to prevent deadlock [53.355–358], or using techniques similar to mutual exclusion in distributed computing [53.359, 360]. These approaches have the benefit of minimizing the planning cost for obtaining a solution. Other, more formal, techniques view the application as a geometric multirobot path planning problem that can be solved precisely in configuration space–time. Chapter 7 includes a discussion of motion planning for multiple robots relevant to this domain. While geometric motion planning approaches provide the most general solutions, they can often be too computationally intensive for practical application, impractical due to the dynamic nature of the environment, or simply unnecessary



Fig. 53.18 Legged robot teams competing in robot soccer

for the problem at hand. In these cases, approximation approaches may be sufficient, such as centralized techniques that limit the search space through roadmapping [53.361, 362], and decoupled approaches that use either prioritized planning [53.363–365] (i. e., generating robot paths one by one) or path coordination (i. e., first planning individual paths for robots, then handling collision avoidance).

53.10.6 Soccer

Since the inception of the RoboCup multirobot soccer domain as a proposed challenge problem for studying coordination and control in multirobot systems [53.366], research in this domain has grown tremendously. This domain incorporates many challenging aspects of multirobot control, including collaboration, robot control architectures, strategy acquisition, real-time reasoning and action, sensor fusion, dealing with adversarial environments, cognitive modeling, and learning. Annual competitions show the ever-improving team capabilities of the robots in a variety of settings, as shown in Fig. 53.18. A key aspect of this domain that is not present in the other multirobot test domains is that robots must operate in *adversarial* environments. This domain is also popular because of its educational benefits, as it brings together students and researchers from across the world in competitions to win the RoboCup challenges. The RoboCup competitions have added an additional search-and-rescue category to the competition [53.367], which has also become a significant area of research (Chap. 66 for more details on this field). Annual proceedings of the RoboCup competitions document much of the research that is incorporated into the multirobot soccer teams. Some representative research works include [53.368–372] (VIDEO 202, VIDEO 209).

53.11 Conclusions and Further Reading

This chapter has surveyed the current state of the art in multirobot systems, examining architectures, communications issues, swarm robot systems, heterogeneous teams, task allocation, learning, and applications. Clearly, significant advances have been made in the field in the last decade. The field is still an active area of research, however, since many open research issues still remain to be solved. Key open research questions remain in the broad areas of system integration, robustness, learning, scalability, generalization, and dealing with heterogeneity.

For example, in the area of system *integration*, an open question is how to effectively allow robot teams to combine a spectrum of approaches toward achieving complete systems that can perform more than a limited set of tasks. We are a long way from creating robust robot networks that can perform physical tasks in the real world. In the area of *robustness*, multirobot teams still need improvements in the ability to degrade gracefully, to reason for fault tolerance, and to achieve complexity without escalating failure rates. The area of *learning* in multirobot teams is still in its infancy, with open questions including how to achieve continual learning in multirobot teams, how to facilitate the use of complex representations, and how to enable humans to influence and/or understand the results of the team learning. *Scalability* is still a challenging problem, in terms of more complex environments as well as ever-larger numbers of robots. We do not yet have a methodology for creating self-organizing robot networks that are robust to labeling (or numbering), with completely decentralized controllers and estimators, and with provable emergent response. This requires basic research at the intersection of control, perception, and communication. Open issues in *generalization* include enabling the robot team to reason about context and increasing the versatility of systems so that they can operate in a variety of different applications. In dealing with *heterogeneity*, open questions include determining theoretical approaches to predicting system performance when all robots are not equal, and determining how to design a robot team optimally for a given application. Advances over the last decade have provided human users with the ability to interact with hundreds or thousands of computers on the Internet. It is necessary to develop similar network-centric approaches to interfacing, both for control and for monitoring. Finally, a major challenge is to create systems that are proactive and anticipate our needs and commands rather than reacting (with delays) to human commands.

For further reading on the topic of multiple mobile robot systems, the reader is referred to survey articles in the field, including [53.2, 288, 373, 374]. Additionally, several special journal issues on this topic have appeared, including [53.1, 375–377]. Some taxonomies of multirobot systems are given in [53.26, 288, 378]. A variety of symposia and workshops have been held on a regular basis on the topic of multirobot systems, in particular the DARS (distributed autonomous robotic systems) series of symposia. Recent proceedings of these workshops and symposia include [53.379–388]. Additional texts on this topic include [53.389–391]. For some excellent further background on networked robotics we direct the reader to [53.31, 35, 46, 52, 55, 75, 90, 104].

Video-References

VIDEO 192 Agents at play: Off-the-shelf software for practical multi-robot applications
available from http://handbookofrobotics.org/view-chapter/53/videodetails/192

VIDEO 193 Handling of a single object by multiple mobile robots based on caster-like dynamics
available from http://handbookofrobotics.org/view-chapter/53/videodetails/193

VIDEO 194 Synchronization and fault detection in autonomous robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/194

VIDEO 195 Self-assembly and morphology control in a swarm-bot
available from http://handbookofrobotics.org/view-chapter/53/videodetails/195

VIDEO 196 CKBOTS reconfigurable robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/196

VIDEO 197 Biologically inspired multi vehicles control algorithm
available from http://handbookofrobotics.org/view-chapter/53/videodetails/197

VIDEO 198 Metamorphic robotic system
available from http://handbookofrobotics.org/view-chapter/53/videodetails/198

VIDEO 199 Multi-robot box pushing
available from http://handbookofrobotics.org/view-chapter/53/videodetails/199

VIDEO 200 Elements of cooperative behavior in autonomous mobile robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/200
VIDEO 201 Coordination of multiple mobile platforms for manipulation and transportation
available from http://handbookofrobotics.org/view-chapter/53/videodetails/201
VIDEO 202 Robots in games and competition
available from http://handbookofrobotics.org/view-chapter/53/videodetails/202
VIDEO 203 A robotic reconnaissance and surveillance team
available from http://handbookofrobotics.org/view-chapter/53/videodetails/203
VIDEO 204 MARS (multiple autonomous robots)
available from http://handbookofrobotics.org/view-chapter/53/videodetails/204
VIDEO 205 A method for transporting a team of miniature robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/205
VIDEO 206 Reconfigurable multi-agents with distributed sensing for robust mobile robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/206
VIDEO 207 Miniature air vehicle cooperative timing missions
available from http://handbookofrobotics.org/view-chapter/53/videodetails/207
VIDEO 208 Distributed manipulation with mobile robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/208
VIDEO 209 Autonomous robot soccer – Through the wormhole with Morgan Freeman
available from http://handbookofrobotics.org/view-chapter/53/videodetails/209
VIDEO 210 A day in the life of a Kiva robot
available from http://handbookofrobotics.org/view-chapter/53/videodetails/210
VIDEO 211 Robot Pebbles – MIT developing self-sculpting smart sand robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/211
VIDEO 212 Transport of a child by swarm-bots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/212
VIDEO 213 Towards a swarm of nano quadrotors
available from http://handbookofrobotics.org/view-chapter/53/videodetails/213
VIDEO 214 Swarm robotics at CU-Boulder
available from http://handbookofrobotics.org/view-chapter/53/videodetails/214
VIDEO 215 Swarm robot system
available from http://handbookofrobotics.org/view-chapter/53/videodetails/215
VIDEO 216 Swarm construction robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/216
VIDEO 217 Multi robot formation control – Khepera team
available from http://handbookofrobotics.org/view-chapter/53/videodetails/217
VIDEO 292 Experiments of escorting a target
available from http://handbookofrobotics.org/view-chapter/53/videodetails/292
VIDEO 293 Formation control via a distributed controller-observer
available from http://handbookofrobotics.org/view-chapter/53/videodetails/293

References

53.1 T. Arai, E. Pagello, L.E. Parker: Editorial: Advances in multi-robot systems, IEEE Trans. Robotics Autom. **18**(5), 655–661 (2002)
53.2 Y. Cao, A. Fukunaga, A. Kahng: Cooperative mobile robotics: Antecedents and directions, Auton. Robots **4**, 1–23 (1997)
53.3 R.G. Brown, J.S. Jennings: A pusher/steerer model for strongly cooperative mobile robot manipulation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 562–568
53.4 D. Milutinović, P. Lima: Modeling and optimal centralized control of a large-size robotic population, IEEE Trans. Robotics **22**(6), 1280–1285 (2006)
53.5 B. Khoshnevis, G.A. Bekey: Centralized sensing and control of multiple mobile robots, Comput. Ind. Eng. **35**(3/4), 503–506 (1998)
53.6 M.J. Matarić: Issues and approaches in the design of collective autonomous agents, Robotics Auton. Syst. **16**, 321–331 (1995)
53.7 E. Ostergaard, G.S. Sukhatme, M.J. Matarić: Emergent bucket brigading, 5th Int. Conf. Auton. Agents (2001)
53.8 M. Matarić: Reinforcement learning in the multi-robot domain, Auton. Robots **4**, 73–83 (1997)
53.9 L.E. Parker: ALLIANCE: An architecture for fault-tolerant multi-robot cooperation, IEEE Trans. Robotics Autom. **14**(2), 220–240 (1998)
53.10 L.E. Parker: Lifelong adaptation in heterogeneous teams: Response to continual variation in individual robot performance, Auton. Robots **8**(3), 239–267 (2000)
53.11 R. Simmons, S. Singh, D. Hershberger, J. Ramos, T. Smith: First results in the coordination of het-

erogeneous robots for large-scale assembly, Proc. ISER 7th Int. Symp. Exp. Robotics (2000)
53.12 J. Deneubourg, S. Goss, G. Sandini, F. Ferrari, P. Dario: Self-organizing collection and transport of objects in unpredictable environments, Jpn.-USA Symp. Flex. Autom. (1990) pp. 1093–1098
53.13 C.R. Kube, H. Zhang: Collective robotics: From social insects to robots, Adapt. Behav. **2**(2), 189–219 (1993)
53.14 R. Beckers, O. Holland, J. Deneubourg: From local actions to global tasks: Stigmergy and collective robotics, Proc. 14th Int. Workshop Synth. Simul. Living Syst. (1994) pp. 181–189
53.15 S. Onn, M. Tennenholtz: Determination of social laws for multi-agent mobilization, Artif. Intell. **95**, 155–167 (1997)
53.16 B.B. Werger: Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams, Artif. Intell. **110**(2), 293–320 (1999)
53.17 M.J. Huber, E. Durfee: Deciding when to commit to action during observation-based coordination, Proc. 1st Int. Conf. Multi-Agent Syst. (1995) pp. 163–170
53.18 H. Asama, K. Ozaki, A. Matsumoto, Y. Ishida, I. Endo: Development of task assignment system using communication for multiple autonomous robots, J. Robotics Mechatron. **4**(2), 122–127 (1992)
53.19 N. Jennings: Controlling cooperative problem solving in industrial multi-agent systems using joint intentions, Artif. Intell. **75**(2), 195–240 (1995)
53.20 M. Tambe: Towards flexible teamwork, J. Artif. Intell. Res. **7**, 83–124 (1997)
53.21 R.T. Vaughan, K. Stoy, G.S. Sukhatme, M.J. Matarić: LOST: Localization-space trails for robot teams, IEEE Trans. Robotics Autom. **18**(5), 796–812 (2002)
53.22 L.E. Parker: The Effect of action recognition and robot awareness in cooperative robotic teams, IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 212–219
53.23 M. Matarić: Behavior-based control: Examples from navigation, learning, and group behavior, J. Exp. Theor. Artif. Intell. **19**(2/3), 323–336 (1997)
53.24 B. MacLennan, G.M. Burghardt: Synthetic ethology and the evolution of cooperative communication, Adapt. Behav. **2**, 161–188 (1993)
53.25 T. Balch, R.C. Arkin: Communiation in reactive multiagent robotic systems, Auton. Robots **1**(1), 27–52 (1995)
53.26 G. Dudek, M. Jenkin, E. Milios, D. Wilkes: A taxonomy for multi-agent robotics, Auton. Robots **3**, 375–397 (1996)
53.27 Z. Butler, K. Kotay, D.L. Rus, M. Vona: Self-reconfiguring robots, Commun. ACM **45**(3), 39–45 (2002)
53.28 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robotic Syst. **13**(11), 755–764 (1996)
53.29 L.E. Parker: The effect of heterogeneity in teams of 100+ mobile robots. In: *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L.E. Parker, F. Schneider (Kluwer, Dordrecht 2003)
53.30 M. Yim, Y. Zhang, D. Duff: Modular robots, IEEE Spectrum **39**(22), 30–34 (2002)
53.31 D. Estrin: *Embedded, Everywhere* (National Academies Press, Washington 2001)
53.32 J. Parrish, S. Viscido, D. Grünbaum: Self-organized fish schools: An examination of emergent properties, Biol. Bull. **202**, 296–305 (2002)
53.33 N. Franks, S. Pratt, E. Mallon, N. Britton, D. Sumpter: Information flow, opinion polling and collective intelligence in house-hunting social insects, Philos. Trans. R. Soc. B **357**, 1567–1584 (2002)
53.34 R.L. Jeanne: Group size, productivity, and information flow in social wasps. In: *Information Processing in Social Insects*, ed. by C. Detrain, J.M. Pasteels, J.L. Deneubourg (Birkhauser, Basel 1999)
53.35 T. Akimoto, N. Hagita: Introduction to a network robot system, IEEE Int. Symp. Intell. Signal Proc. Comm. (2006)
53.36 Argo Floats: *A global array of 3000 Free-Drifting Profiling Floats for Environmental Monitoring* (Argo Information Center, Ramonville 2007)
53.37 G.S. Sukhatme, A. Dhariwal, B. Zhang, C. Oberg, B. Stauffer, D.A. Caron: The design and development of a wireless robotic networked aquatic microbial observing system, Environ. Eng. Sci. **24**(2), 205–215 (2006)
53.38 W. Kaiser, G. Pottie, M. Srivastava, G.S. Sukhatme, J. Villasenor, D. Estrin: Networked infomechanical systems (NIMS) for ambient intelligence. In: *Ambient Intelligence*, ed. by W. Weber, J.M. Rabaey, E. Aarts (Springer, Berlin, Heidelberg 2005)
53.39 Amarss: *Networked Minirhizotron Planning and Initial Deployment* (Center for Embedded Networkedsensing, Los Angeles 2007)
53.40 H. Singh, J. Catipovic, R. Eastwood, L. Freitag, H. Henriksen, F.F. Hover, D. Yoerger, J.G. Bellingham, B.A. Moran: An integrated approach to multiple AUV communications, navigation and docking, MTS/IEEE Conf. Proc. OCEANS (1996) pp. 59–64
53.41 I. Vasilescu, M. Dunbabin, P. Corke, K. Kotay, D. Rus: Data collection, storage, and retrieval with an underwater sensor network, Proc. ACM Sens. Syst. (2005)
53.42 N. Leonard, D. Paley, F. Lekien, R. Sepulchre, D.M. Fratantoni, R. Davis: Collective motion, sensor networks and ocean sampling, Proc. IEEE **95**(1), 48–74 (2006)
53.43 D.O. Popa, A.S. Sanderson, R.J. Komerska, S.S. Mupparapu, D.R. Blidberg, S.G. Chappell: Adaptive sampling algorithms for multiple autonomous underwater vehicles, IEEE/OES AUV2004: A Workshop Multiple Auton. Underw. Veh. Oper. (2004)
53.44 L. Chaimowicz, A. Cowley, B. Grocholsky, M.A. Hsieh, J.F. Keller, V. Kumar, C.J. Taylor: Deploying air-ground multi-robot teams in urban environments, 3rd Multi-Robot Syst. Workshop (2005)

Avidbots Ex1016
Page 1392 of 2154

53.45 B. Grocholsky, R. Swaminathan, J. Keller, V. Kumar, G. Pappas: Information driven coordinated air-ground proactive sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

53.46 A. Howard, L.E. Parker, G.S. Sukhatme: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection, Int. J. Robotics Res. **25**(5/6), 431–447 (2006)

53.47 D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, B. Stewart: Distributed multirobot exploration and mapping, Proc. IEEE **94**(7), 1325–1339 (2006)

53.48 M.A. Hsieh, A. Cowley, J.F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C.J. Talyor, Y. Endo, R. Arkin, B. Jung, D. Wolf, G. Sukhatme, D.C. MacKenzie: Adaptive teams of autonomous aerial and ground robots for situational awareness, J. Field Robotics **24**(11/12), 991–1014 (2007)

53.49 J. Seyfied, M. Szymanski, N. Bender, R. Estana, M. Theil, H. Worn: The I-swarm project: Intelligent small world autonomous robots for micromanipulation, SAB 2004 Int. Workshop (2004)

53.50 F. Mondada, G.C. Pettinaro, A. Guignard, I.W. Kwee, D. Floreano, J.-L. Deneubourg, S. Nofli, L.M. Gambardella, M. Dorigo: SwarmBot: A new distributed robotic concept, Auton. Robots **17**, 193–221 (2004)

53.51 A. Ollero, S. Lacroix, L. Merino, J. Gancet, J. Wiklund, V. Remuss, I. Veiga, L.G. Gutierrez, D.X. Viegas, M.A. Gonzalez, A. Mallet, R. Alami, R. Chatila, G. Hommel, F.J. Colmenero, B. Arrue, J. Ferruz, R. Martinez de Dios, F. Caballero: Architecture and perception issues in the COMETS multi-UAV project, IEEE Robotics Autom. Mag. **12**(2), 46–57 (2005)

53.52 Q. Li, D. Rus: Navigation protocols in sensor networks, ACM Trans. Sens. Netw. **1**(1), 3–35 (2005)

53.53 R. Bajcsy: Active perception, Proc. IEEE **76**, 996–1005 (1988)

53.54 T. Eren, D. Goldenberg, W. Whitley, Y.R. Yang, S. Morse, B.D.O. Anderson, P.N. Belhumeur: Rigidity, computation, and randomization of network localization, Proc. IEEE INFOCOM (2004)

53.55 A. Hsieh, A. Cowley, V. Kumar, C.J. Taylor: Towards the deployment of a mobile robot network with end-to-end performance guarantees, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

53.56 B. Grocholsky, S. Bayraktar, V. Kumar, C.J. Taylor, G. Pappas: Synergies in feature localization by air-ground teams, Proc. 9th Int. Symp. Exp. Robotics (2004)

53.57 B. Grocholsky, E. Stump, V. Kumar: An extensive representation for range-only SLAM, Int. Symp. Exp. Robotics (2006)

53.58 A. Howard, M.J. Matarić, G.S. Sukhatme: An incremental self-deployment algorithm for mobile sensor networks, Auton. Robots **13**(2), 113–126 (2002)

53.59 Z. Butler, D. Rus: Controlling mobile networks for monitoring events with coverage constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

53.60 M. Chu, J. Reich, F. Zhao: Distributed attention for large video sensor networks, Intell. Distrib. Surveill. Syst. Semin (2004)

53.61 B. Jung, G.S. Sukhatme: Tracking targets using multiple robots: The effect of environment occlusion, Auton. Robots **13**(3), 191–205 (2002)

53.62 R.M. Murray, K.J. Åström, S.P. Boyd, R.W. Brockett, G. Stein: Future directions in control in an information-rich world, IEEE Control Syst. Mag. (2003)

53.63 A. Pant, P. Seiler, K. Hedrick: Mesh stability of look-ahead interconnected systems, IEEE Trans. Autom. Control **47**, 403–407 (2002)

53.64 T. Sugar, J. Desai, V. Kumar, J.P. Ostrowski: Coordination of multiple mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3022–3027

53.65 R.W. Beard, J. Lawton, F.Y. Hadaegh: A coordination architecture for spacecraft formation control, IEEE Trans. Control Syst. Technol. **9**, 777–790 (2001)

53.66 A. Das, J. Spletzer, V. Kumar, C. Taylor: Ad hoc networks for localization and control, Proc. IEEE Conf. Decis. Control (2002) pp. 2978–2983

53.67 J. Manyika, H. Durrant-Whyte: *Data Fusion and Sensor Management: An Information-Theoretic Approach* (Prentice Hall, Upper Saddle River 1994)

53.68 H.G. Tanner, A. Jadbabaie, G.J. Pappas: Stable flocking of mobile agents, Part I: Fixed topology, Proc. IEEE Conf. Decis. Control (2003) pp. 2010–2015

53.69 J.M. Fowler, R. D'Andrea: Distributed control of close formation flight, Proc. IEEE Conf. Decis. Control (2002) pp. 2972–2977

53.70 J.P. Desai, J.P. Ostrowski, V. Kumar: Modeling and control of formations of nonholonomic mobile robots, IEEE Trans. Robotics Autom. **17**(6), 905–908 (2001)

53.71 H.G. Tanner, V. Kumar, G.J. Pappas: Leader-to-formation stability, IEEE Trans. Robotics Autom. **20**(3), 443–455 (2004)

53.72 S. Loizou, V. Kumar: Relaxed input to state stability properties for navigation function based systems, Proc. IEEE Conf. Decis. Control (2006)

53.73 M. Batalin, G.S. Sukhatme: Coverage, exploration and deployment by a mobile robot and communication network, Telecommun. Syst. J. **26**(2), 181–196 (2004)

53.74 K.J. O'hara, V. Bigio, S. Whitt, D. Walker, T.R. Balch: Evaluation of a large scale pervasive embedded network for robot path planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2072–2077

53.75 V. Kumar, N. Leonard, A.S. Morse (Eds.): *Cooperative control*, Lecture Notes in Control and Information Sciences, Vol. 309 (Springer, Berlin, Heidelberg 2004)

53.76 J. Chen, S. Teller, H. Balakrishnan: Pervasive pose-aware applications and infrastructure, IEEE Comput. Graph. Appl. **23**(4), 14–18 (2003)

53.77 A. Savvides, C.-C. Han, M. Srivastava: Dynamic fine-grained localization in ad-hoc networks of sensors, Proc. 7th Annu. Int. Conf. Mobile Comput. Netw. (MOBICOM-01) (2001) pp. 166–179

Avidbots Ex1016
Page 1393 of 2154

53.78 D. Moore, J. Leonard, D. Rus, S.J. Teller: Robust distributed network localization with noisy range measurements, Proc. 2nd Int. Conf. Embed. Netw. Sens. Syst. (SenSys) (2004) pp. 50–61

53.79 C. Detweiler, J. Leonard, D. Rus, S. Teller: Passive mobile robot localization within a fixed beacon field, Proc. Int. Workshop Algorithmic Found. Robotics (2006)

53.80 N. Bulusu, J. Heidemann, D. Estrin: Adaptive beacon placement, Proc. 21st Int. Conf. Distrib. Comput. Syst. (ICDCS-01) (2001) pp. 489–498

53.81 S.N. Simic, S. Sastry: Distributed Localization in Wireless ad hoc Networks, Tech. Rep. UCB/ERL M02/26 (2001), http://www.eecs.berkeley.edu/Pubs/TechRpts/2002/4010.html

53.82 P. Corke, R. Peterson, D. Rus: Communication-assisted localization and navigation for networked robots, Int. J. Robotics Res. **4**(9), 116 (2005)

53.83 R. Nagpal, H.E. Shrobe, J. Bachrach: Organizing a global coordinate system from local information on an ad hoc sensor network, Lect. Notes Comput. Sci. **2634**, 333–348 (2003)

53.84 R.K. Williams, A. Gasparri, A. Priolo, G.S. Sukhatme: Evaluating network rigidity in realistic systems: Decentralization, asynchronicity, and parallelization, IEEE Trans. Robotics **30**(4), 950–965 (2014)

53.85 A. Howard, M.J. Matarić, G.S. Sukhatme: Putting the 'i' in 'team': An ego-centric approach to cooperative localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 868–892

53.86 A. Howard, M.J. Matarić, G.S. Sukhatme: Localization for mobile robot teams using maximum likelihood estimation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2002) pp. 434–459

53.87 R. Vidal, O. Shakernia, H.J. Kim, D.H. Shim, S. Sastry: Probabilistic pursuit-evasion games: Theory, implementation and experimental evaluation, IEEE Trans. Robotics Autom. **18**(5), 662–669 (2002)

53.88 M. Batalin, M.H. Rahimi, Y. Yu, D. Liu, A. Kansal, G.S. Sukhatme, W. Kaiser, M. Hansen, G. Pottie, M. Srivastava, D. Estrin: Call and response: Experiments in sampling the environment, Proc. 2nd Int. Conf. Embed. Netw. Sens. Syst. (SenSys) (2004) pp. 25–38

53.89 M.H. Rahimi, W. Kaiser, G.S. Sukhatme, D. Estrin: Adaptive sampling for environmental field estimation using robotic sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2005) pp. 747–753

53.90 F. Zhao, L. Guibas: *Wireless Sensor Networks: An Information Processing Approach* (Morgan Kaufmann, New York 2004)

53.91 F. Zhang, B. Grocholsky, V. Kumar: Formations for localization of robot networks, IEEE Int. Conf. Robotics Autom. (2004)

53.92 R.K. Williams, G.S. Sukhatme: Constrained interaction and coordination in proximity-limited multiagent systems, IEEE Trans. Robotics **29**(4), 930–944 (2013)

53.93 J. Cortes, S. Martinez, T. Karatas, F. Bullo: Coverage control for mobile sensing networks, IEEE Trans. Robotics Autom. **20**(2), 243–255 (2004)

53.94 J. Cortes, S. Martinez, F. Bullo: Spatially-distributed coverage optimization and control with limited-range interactions, ESAIM Control Optim. Calc. Var. **11**, 691–719 (2005)

53.95 M. Schwager, J. McLurkin, D. Rus: Distributed coverage control with sensory feedback for networked robots, Proc. Robotics Sci. Syst. (RSS) (2006)

53.96 M. Schwager, J.-J. Slotine, D. Rus: Decentralized adaptive control for coverage for networked robots, Proc. Int. Conf. Robotics Autom. (2007)

53.97 A. Krause, C. Guestrin, A. Gupta, J. Kleinberg: Near-optimal sensor placements: Maximizing information while minimizing communication cost, 5th Int. Conf. Inf. Process Sens. Netw. (IPSN) (2006)

53.98 V. Chvatal: A combinatorial theorem in plane geometry, J. Comb. Theory Ser. **18**, 39–41 (1975)

53.99 J. O'Rourke: *Art Gallery Theorems and Algorithms* (Oxford Univ. Press, New York 1987)

53.100 S. Fisk: A short proof of Chvatal's watchmen theorem, J. Comb. Theory Ser. **24**, 374 (1978)

53.101 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Autom. Control **48**(6), 988–1001 (2003)

53.102 S. Sinha, M. Pollefeys: Camera network calibration from dynamic silhouettes, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2004)

53.103 R. Collins, A. Lipton, H. Fujiyoshi, T. Kanade: Algorithms for cooperative multisensor surveillance, Proc. IEEE **89**(10), 1456–1477 (2001)

53.104 A. Kansal, W. Kaiser, G. Pottie, M. Srivastava, G.S. Sukhatme: Reconfiguration methods for mobile sensor networks, ACM Trans. Sens. Networks **3**(4), 1–28 (2007)

53.105 D. Payton, M. Daily, R. Estkowski, M. Howard, C. Lee: Pheromone robotics, Auton. Robots **11**, 319–324 (2001)

53.106 A. Howard, M.J. Matarić, G.S. Sukhatme: Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem, Proc. Int. Symp. Distrib. Auton. Robotic Syst. (2002) pp. 299–308

53.107 S. Poduri, G.S. Sukhatme: Constrained coverage for mobile sensor networks, IEEE Int. Conf. Robotics Autom. (2004) pp. 165–172

53.108 R. Wattenhofer, L. Li, P. Bahl, Y.M. Wang: A cone-based distributed topology-control algorithm for wireless multi-hop networks, IEEE/ACM Trans. Netw. **13**(1), 147–159 (2005)

53.109 H. Zhang, J.C. Hou: On deriving the upper bound of alphalifetime for large sensor networks, ACM Trans. Sens. Netw. **1**(6), 272–300 (2005)

53.110 S. Poduri, S. Pattem, B. Krishnamachari, G.S. Sukhatme: Using local geometry for tunable topology control in sensor networks, IEEE Trans. Mobile Comput. **8**(2), 218–230 (2009)

53.111 M.A. Hsieh, V. Kumar: Pattern generation with multiple robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

53.112 R.N. Smith, M. Schwager, S.L. Smith, B.H. Jones, D. Rus, G.S. Sukhatme: Persistent ocean mon-

itoring with underwater gliders: Adapting spatiotemporal sampling resolution, J. Field Robotics **28**(5), 714–741 (2011)
53.113 J. Das, F. Py, T. Maughan, T. O'Reilly, M. Messié, J. Ryan, G.S. Sukhatme, K. Rajan: Coordinated sampling of dynamic oceanographic features with AUVs and drifters, Int. J. Robotics Res. **31**(5), 626–646 (2012)
53.114 G.A. Hollinger, S. Choudhary, P. Qarabaqi, C. Murphy, U. Mitra, G.S. Sukhatme, M. Stojanovic, H. Singh, F. Hover: Underwater data collection using robotic sensor networks, IEEE J. Sel. Areas Commun. **30**(5), 899–911 (2012)
53.115 F. Ye, G. Zhong, J. Cheng, L. Zhang, S. Lu: Peas: A robust energy conserving protocol for long-lived sensor networks, Int. Conf. Distrib. Comput. Syst. (2003)
53.116 X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, C. Gill: Integrated coverage and connectivity configuration in wireless sensor networks, 1st ACM Conf. Embed. Netw. Sens. Syst. (SenSys) (2003)
53.117 D. Tian, N.D. Georganas: Connectivity maintenance and coverage preservation in wireless sensor networks, Ad Hoc Networks **3**(6), 744–761 (2005)
53.118 G. Theraulaz, S. Goss, J. Gervet, J.-L. Deneubourg: Task differentiation in Polistes wasp colonies: A model for self-organizing groups of robots, Proc. 1st Int. Conf. Simul. Adapt. Behav. (1990) pp. 346–355
53.119 L. Steels: Cooperation Between Distributed Agents Through Self-Organization, Proc. IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990)
53.120 A. Drogoul, J. Ferber: From tom thumb to the dockers: Some experiments with foraging robots, Proc. 2nd Int. Conf. Simul. Adapt. Behav. (1992) pp. 451–459
53.121 M.J. Matarić: Designing emergent behaviors: From local interactions to collective intelligence, Proc. 2nd Int. Conf. Simul. Adapt. Behav. (1992) pp. 432–441
53.122 G. Beni, J. Wang: Swarm intelligence in cellular robotics systems, Proc. NATO Adv. Workshop Robots Biol. Syst. (1989)
53.123 D. Stilwell, J. Bay: Toward the development of a material transport system using swarms of ant-like robots, Proc. IEEE Int. Conf. Robotics Autom. (1993) pp. 766–771
53.124 T. Fukuda, S. Nakagawa, Y. Kawauchi, M. Buss: Self organizing robots based on cell structures – CEBOT, Proc. IEEE Int. Workshop Intell. Robots Syst. (1988) pp. 145–150
53.125 J.H. Reif, H.Y. Wang: Social Potential fields: A distributed behavior control for autonomous robots, Robotics Auton. Syst. **27**(3), 171–194 (1999)
53.126 L.E. Parker: Designing control laws for cooperative agent teams, Proc. IEEE Int. Cont. Robotics Autom. (1993) pp. 582–587
53.127 K. Sugihara, I. Suzuki: Distributed algorithms for formation of goemetric patterns with many mobile robots, J. Robotic Syst. **13**(3), 127–139 (1996)
53.128 V. Gazi: Swarm aggregations using artificial potentials and sliding-mode control, IEEE Trans. Robotics **21**(6), 1208–1214 (2005)
53.129 J. McLurkin, J. Smith: Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots, Symp. Distrib. Auton. Robots Syst. (2004)
53.130 D. Gage: Randomized search strategies with imperfect sensors, Proc. SPIE Mobile Robots VIII (1993) pp. 270–279
53.131 T. Balch: The impact of diversity on performance in robot foraging, Proc. 3rd Ann. Conf. Auton. Agents (1999) pp. 92–99
53.132 Z.J. Butler, A.A. Rizzi, R.L. Hollis: Cooperative coverage of rectilinear environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000)
53.133 A.I. Mourikis, S.I. Roumeliotis: Performance analysis of multirobot cooperative localization, IEEE Trans. Robotics **22**(4), 666–681 (2006)
53.134 R. Grabowski, L.E. Navarro-Serment, C.J. Paredis, P.K. Khosla: Heterogeneous teams of modular robots for mapping and exploration, Auton. Robots **8**(3), 271–298 (2000)
53.135 M. Berhault, H. Huang, P. Keskinocak, S. Koenig, W. Elmaghraby, P. Griffin, A. Kleywegt: Robot exploration with combinatorial auctions, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1957–1962
53.136 J. Kim, J.M. Esposito, V. Kumar: An RRT-based algorithm for testing and validating mulit-robot controllers, Proc. Robotics Sci. Syst. I (2005)
53.137 Z. Cao, M. Tin, L. Li, N. Gu, S. Wang: Cooperative hunting by distributed mobile robots based on local interaction, IEEE Trans. Robotics **22**(2), 403–407 (2006)
53.138 R.W. Beard, T.W. McLain, M. Goodrich: Coordinated target assignment and intercept for unmanned air vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2581–2586
53.139 J. Clark, R. Fierro: Cooperative hybrid control of robotic sensors for perimeter detection and tracking, Proc. Am. Control Conf. (2005) pp. 3500–3505
53.140 P. Stone, M. Veloso: A layered approach to learning client behaviors in the RoboCup soccer server, Appl. Artif. Intell. **12**, 165–188 (1998)
53.141 J. McLurkin: Stupid Robot Tricks: Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots, Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 2004)
53.142 F. Mondada, L.M. Gambardella, D. Floreano, S. Nolfi, J.L. Deneuborg, M. Dorigo: The cooperation of swarm-bots: Physical interactions in collective robotics, IEEE Robotics Autom. Mag. **12**(2), 21–28 (2005)
53.143 T.W. Mather, M.A. Hsieh: Macroscopic modeling of stochastic deployment policies with time delays for robot ensembles, Int. J. Robotics Res. **30**(5), 590–600 (2011)
53.144 X.C. Ding, M. Kloetzer, Y. Chen, C. Belta: Automatic deployment of robot teams, IEEE Robotics Autom. Mag. **18**(3), 75–86 (2022)

Avidbots Ex1016
Page 1395 of 2154

53.145 F.S. Melo, M. Veloso: Decentralized MDPs with sparse interactions, Artif. Intell. **175**(11), 1757–1789 (2011)

53.146 P. Tsiotras, L.I.R. Castro: Extended multi-agent consensus protocols for the generation of geometric patterns in the plane, Am. Controls Conf. (2011) pp. 3850–3855

53.147 K. Cheng, P. Dasgupta: Weighted voting game based multi-robot team formation for distributed area coverage, Proc. 3rd Int. Symp. Pract. Cognit. Agents Robots (2010) pp. 9–15

53.148 A.A. Taheri, M. Afshar, M. Asadpour: Influence maximization for informed agents in collective behavior, Distrib. Auton. Robotic Syst. (2013) pp. 389–402

53.149 M.A. Hsieh, A. Halász, E.D. Cubuk, S. Schoenholz, A. Martinoli: Specialization as an optimal strategy under varying external conditions, Proc. IEEE Int. Conf. Robotics Autom. (2009) pp. 1941–1946

53.150 N. Hoff, R. Wood, R. Nagpal: Distributed colony-level algorithm switching for robot swarm foraging, Distrib. Auton. Robotic Syst. (2013) pp. 417–430

53.151 A. Winfield, J. Nembrini: Safety in numbers: Fault-tolerance in robot systems, Int. J. Model. Identif. Control **1**(1), 30–37 (2006)

53.152 T. Fukuda, S. Nakagawa: Dynamically reconfigurable robotic system, IEEE Int. Conf. Robotics Autom. (1988) pp. 1581–1586

53.153 M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, G.S. Chirikjian: Modular self-reconfigurable robot systems: Challenges and opportunities for the future, IEEE Robotics Autom. Mag. **14**(1), 43–52 (2007)

53.154 K. Gilpin, D. Rus: Modular robot systems: From self-assembly to self-disassembly, IEEE Robotics Autom. Mag. **17**(3), 38–53 (2010)

53.155 M. Yim: A reconfigurable modular robot with many modes of locomotion, JSME Int. Conf. Adv. Mechatron. (1993) pp. 283–288

53.156 M. Yim: New locomotion gaits, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 2508–2514

53.157 A. Castano, P. Will: Mechanical design of a module for reconfigurable robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 2203–2209

53.158 W.-M. Shen, P. Will: Docking in self-reconfigurable robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1049–1054

53.159 A. Castano, A. Behar, P. Will: The conro modules for reconfigurable robots, IEEE Trans. Mechatron. **7**(4), 403–409 (2002)

53.160 S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, S. Kokaji: M-TRAN: Self-reconfigurable modular robotic system, IEEE/ASME Trans. Mechatron. **7**(4), 431–441 (2002)

53.161 A. Kamimura, H. Kurokawa, E. Yoshida, S. Murata, K. Tomita, S. Kokaji: Automatic locomotion design and experiments for a modular robotic system, IEEE/ASME Trans. Mechatron. **10**(3), 314–325 (2005)

53.162 H. Kurokawa, K. Tomita, A. Kamimura, E. Yoshida, S. Kokahji, S. Murata: Distributed self-reconfiguration control of modular robot M-TRAN, IEEE Int. Conf. Mechatron. Autom. (2005) pp. 254–259

53.163 S. Murata, K. Kakomura, H. Kurokawa: Docking experiments of a modular robot by visual feedback, Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS) (2006) pp. 625–630

53.164 D. Marbach, A.J. Ijspeert: Online optimization of modular robot locomotion, IEEE Int. Conf. Mechatron. Autom. (2005) pp. 248–253

53.165 E.H. Østergaard, H.H. Lund: Evolving control for modular robotic units, IEEE Int. Symp. Comput. Intell. Robotics Autom. (2003) pp. 886–892

53.166 M.W. Jørgensen, E.H. Østergaard, H.H. Lund: Modular ATRON: Modules for a self-reconfigurable robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2068–2073

53.167 B. Salemi, M. Moll, W.-M. Shen: SUPERBOT: A deployable, multi-functional, and modular self-reconfigurable robotic system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 3636–3641

53.168 M. Yim, D.G. Duff, K.D. Roufas: PolyBot: A modular reconfigurable robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 514–520

53.169 M. Yim, Y. Zhang, K. Roufas, D. Duff, C. Eldershaw: Connecting and disconnecting for self–reconfiguration with PolyBot, IEEE/ASME Trans. Mechatron. **7**(4), 442–451 (2003)

53.170 M. Yim, B. Shirmohammadi, J. Sastra, M. Park, M. Dugan, C.J. Taylor: Towards robotic self-reassembly after explosion, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2007) pp. 2767–2772

53.171 V. Zykov, E. Mytilinaios, M. Desnoyer, H. Lipson: Evolved and designed self-reproducing modular robotics, IEEE Trans. Robotics **23**(2), 308–319 (2007)

53.172 P.J. White, M.L. Posner, M. Yim: Strength analysis of miniature folded right angle tetrahedron chain programmable matter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2785–2790

53.173 G.S. Chirikjian: Kinematics of a metamorphic robotic system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 449–455

53.174 G. Chirikjian, A. Pamecha, I. Ebert-Uphoff: Evaluating efficiency of self-reconfiguration in a class of modular robots, J. Robotic Syst. **13**(5), 317–388 (1996)

53.175 A. Pamecha, I. Ebert-Uphoff, G.S. Chirikjian: Useful metrics for modular robot motion planning, IEEE Trans. Robotics Autom. **13**(4), 531–545 (1997)

53.176 J.E. Walter, E.M. Tsai, N.M. Amato: Algorithms for fast concurrent reconfiguration of hexagonal metamorphic robots, IEEE Trans. Robotics **21**(4), 621–631 (2005)

53.177 S. Murata, H. Kurokawa, S. Kokaji: Self-assembling machine, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 441–448

53.178 E. Yoshida, S. Murata, K. Tomita, H. Kurokawa, S. Kokaji: Distributed formation control for a modular mechanical system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 1090–1097

Avidbots Ex1016
Page 1396 of 2154

53.179 H. Kurokawa, S. Murata, E. Yoshida, K. Tomita, S. Kokaji: A 3-D self-reconfigurable structure and experiments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 860–865

53.180 E. Yoshida, S. Murata, S. Kokaji, A. Kamimura, K. Tomita, H. Kurokawa: Get back in shape! A hardware prototype self-reconfigurable modular microrobot that uses shape memory alloy, IEEE Robotics Autom. Mag. **9**(4), 54–60 (2002)

53.181 J. Gargus, B. Kim, I. Llamas, J. Rossignac, C. Shaw: Finger Sculpting with Digital Clay, Tech. Rep. GIT-GVU-02-22 (2002)

53.182 K. Kotay, D. Rus, M. Vona, C. McGray: The self-reconfiguring robotic molecule, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 424–431

53.183 K. Kotay, D. Rus: Motion synthesis for the self-reconfiguring robotic molecule, IEEE Int. Conf. Intell. Robots Syst. (1998) pp. 843–851

53.184 K. Kotay, D. Rus: Algorithms for self-reconfiguring molecule motion planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000)

53.185 Z.J. Butler, K. Kotay, D. Rus, K. Tomita: Generic decentralized control for lattice-based self-reconfigurable robots, Int. J. Robotics Res. **23**(9), 919–937 (2004)

53.186 D. Rus, M. Vona: A basis for self-reconfiguring robots using crystal modules, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 2194–2202

53.187 D. Rus, M. Vona: Crystalline robots: Self-reconfiguration with compressible unit modules, Int. J. Robotics Res. **22**(9), 699–715 (2003)

53.188 J.W. Suh, S.B. Homans, M. Yim: Telecubes: Mechanical design of a module for self-reconfigurable robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 4095–4101

53.189 C.-J. Chiang, G.S. Chirikjian: Modular robot motion planning using similarity metrics, Auton. Robots **10**, 91–106 (2001)

53.190 M. Koseki, K. Minami, N. Inou: Cellular robots forming a mechanical structure (evaluation of structural formation and hardware design of CHOBIE II), Proc. 7th Int. Symp. Distrib. Auton. Robotic Syst. (DARS) (2004) pp. 131–140

53.191 B.K. An: Em-cube: Cube-shaped, self-reconfigurable robots sliding on structure surfaces, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3149–3155

53.192 C. Ünsal, P.K. Khosla: Mechatronic design of a modular self-reconfiguring robotic system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 1742–1747

53.193 K.C. Prevas, C. Ünsal, M.Ö. Efe, P.K. Khosla: A hierarchical motion planning strategy for a uniform self-reconfigurable modular robotic system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 787–792

53.194 K. Hosokawa, T. Tsujimori, T. Fujii, H. Kaetsu, H. Asama, Y. Kuroda, I. Endo: Self-organizing collective robots with morphogenesis in a vertical plane, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 2858–2863

53.195 S. Goldstein, J. Campbell, T. Mowry: Programmable Matter, IEEE Computer **38**(6), 99–101 (2005)

53.196 S.C. Goldstein, J.D. Campbell, T.C. Mowry: Programmable matter, Computer **38**(6), 99–101 (2005)

53.197 M.E. Karagozler, S.C. Goldstein, J.R. Reid: Stress-driven MEMS assembly + electrostatic forces = 1 mm diameter robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 2763–2769

53.198 P. Pillai, J. Campbell, G. Kedia, S. Moudgal, K. Sheth: A 3-D fax machine based on claytronics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4728–4735

53.199 P. White, K. Kopanski, H. Lipson: Stochastic self-reconfigurable cellular robotics, IEEE Conf. Robotics Autom. (2004) pp. 2888–2893

53.200 P. White, V. Zykov, J. Bongard, H. Lipson: Three dimensional stochastic reconfiguration of modular robots, Robotics Sci. Syst. (2005)

53.201 M. Tolley, J. Hiller, H. Lipson: Evolutionary design and assembly planning for stochastic modular robots, Proc. IEEE/RSJ. Int. Conf. Intell. Robotics Syst. (IROS) (2009) pp. 73–78

53.202 M. Tolley, H. Lipson: Fluidic manipulation for scalable stochastic 3-D assembly of modular robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2473–2478

53.203 M.T. Tolley, H. Lipson: Programmable 3-D stochastic fluidic assembly of cm-scale modules, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4366–4371

53.204 K. Gilpin, K. Kotay, D. Rus, I. Vasilescu: Miche: Modular shape formation by self-disassembly, Int. J. Robotics Res. **27**, 345–372 (2008)

53.205 R. Oung, F. Bourgault, M. Donovan, R. D'Andrea: The distributed flight array, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

53.206 G.J. Hamlin, A.C. Sanderson: Tetrobot: A modular system for hyper-redundant parallel robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 154–159

53.207 A. Lyder, R.F.M. Garcia, K. Stoy: Mechanical design of Odin, an extendable heterogeneous deformable modular robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 883–888

53.208 A. Lyder, H.G. Peterson, K. Stoy: Representation and shape estimation of Odin, a parallel under-actuated modular robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 5275–5280

53.209 C.-H. Yu, K. Haller, D. Ingber, R. Nagpal: Morpho: A self-deformable modular robot inspired by cellar structure, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3571–3578

53.210 M. Shimizu, A. Ishiguro, T. Kawakatsu: A modular robot that exploits a spontaneous connectivity control mechanism, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 1899–1904

53.211 M. Shimizu, T. Mori, A. Ishiguro: A Development of a modular robot that enables adaptive reconfiguration, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 174–179

Avidbots Ex1016
Page 1397 of 2154

53.212 S. Funiak, P. Pillai, M.P. Ashley-Rollman, J.D. Campbell, S.C. Goldstein: Distributed localization of modular robot ensembles, Int. J. Robotics Res. **28**(8), 946–961 (2009)

53.213 M. Rubenstein, W.-M. Shen: Scalable self-assembly and self-repair in a collective of robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1484–1489

53.214 M. Rubenstein, W.-M. Shen: Automatic scalable size selection for the shape of a distributed robotic collective, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 508–513

53.215 M. Rubenstein, C. Ahler, R. Nagpal: Kilobot: A low cost scalable robot system for collective behaviors, IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

53.216 E. Hawkes, B. An, N.M. Benbernou, H. Tanaka, S. Kim, E.D. Demaine, D. Rus, R.J. Wood: Programmable matter by folding, Proc. Natl. Acad. Sci. USA **107**(28), 12441–12445 (2010)

53.217 B. An, D. Rus: Programming and controlling self-folding sheets, IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

53.218 G. Whitesides, B. Grzybowski: Self-assembly at all scales, Sci. USA **295**, 2418–2421 (2002)

53.219 G.M. Whitesides, M. Boncheva: Beyond molecules: Self-assembly of mesoscope and macroscopic components, Proc. Natl. Acad. Sciences **99**(8), 4769–4774 (2002)

53.220 D.H. Garcias, J. Tien, T.L. Breen, C. Hsu, G.M. Whitesides: Forming electrical networks in three dimensions by self-assembly, Science **289**(5482), 1170–1172 (2000)

53.221 S. Miyashita, M. Kessler, M. Lungarella: How morphology affects self-assembly in a stochastic modular robot, IEEE Int. Conf. Robotics Autom. (2008) pp. 3533–3538

53.222 K. Hosokawa, I. Shimoyama, H. Miura: Dynamics of self-assembling systems: Analogy with chemical kinematics, Artif. Life **1**(4), 413–427 (1994)

53.223 M. Shimizu, K. Suzuki: A Self-repairing structure for modules and its control by vibrating actuation mechanisms, IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 4281–4286

53.224 P.W.K. Rothemund, E. Winfree: The program-size complexity of self-assembled squares, 32rd Annu. ACM Symp. Theory Comput. (2000) pp. 459–468

53.225 L. Adleman, Q. Cheng, A. Goel, M.-D. Huang: Running time and program size for self-assembled squares, 33rd Annu. ACM Symp. Theory Comput. (2001) pp. 740–748

53.226 G. Aggarwal, M.H. Goldwasser, M.-Y. Kao, R.T. Schweller: Complexities for generalized models of self-assembly, 15th Annu. ACM-SIAM Symp. Discrete Algorithms (2004) pp. 880–889

53.227 J. Bishop, S. Burden, E. Klavins, R. Kreisberg, W. Malone, N. Napp, T. Nguyen: Programmable parts: A demonstration of the grammatical approach to self-organization, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 3684–3691

53.228 S. Griffith, D. Goldwater, J.M. Jacobson: Robotics: Self-replication from random parts, Nature **437**, 636 (2005)

53.229 C. Jones, M.J. Matarić: From local to global behavior in intelligent self-assembly, IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 721–726

53.230 J. Kelly, H. Zhang: Combinatorial optimization of sensing for rule-based planar distributed assembly, IEEE Int. Conf. Intell. Robots Syst. (2006) pp. 3728–3734

53.231 J. Werfel: Anthills Built to Order: Automating Construction with Artificial Swarms, Ph.D. Thesis (MIT, Cambridge 2006)

53.232 B. Donald, C.G. Levey, C.D. McGray, I. Paprotny, D. Rus: An untethered, electrostatic, globally controllable MEMS micro-robot, J. Microelectromech. Syst. **15**(1), 1–15 (2006)

53.233 B.R. Donald, C.G. Levey, I. Paprotny: Planar microassembly by parallel actuator of MEMS microrobots, J. Microelectromech. Syst. **17**(4), 789–808 (2008)

53.234 C. Pawashe, S. Floyd, M. Sitti: Assembly and disassembly of magnetic mobile micro-robots towards 2-D reconfigurable micro-systems, Int. Symp. Robotics Res. (2009)

53.235 N. Napp, S. Burden, E. Klavins: The statistical dynamics of programmed self-assembly, IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1469–1476

53.236 K. Gilpin, D. Rus: A distributed algorithm for 2-D shape duplication with smart pebble robots, IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

53.237 K. Gilpin, A. Knaian, D. Rus: Robot pebbles: One centimeter robotic modules for programmable matter through self-disassembly, IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

53.238 L.E. Parker: The effect of heterogeneity in teams of 100+ mobile robots. In: *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L.E. Parker, F. Schneider (Kluwer, Dordrecht 2003)

53.239 T. Balch: Hierarchic social entropy: An information theoretic measure of robot team diversity, Auton. Robots **8**(3), 209–238 (2000)

53.240 D. Jung, A. Zelinsky: Grounded symbolic communication between heterogeneous cooperating robots, Auton. Robots **8**(3), 269–292 (2000)

53.241 R.R. Murphy: Marsupial robots for urban search and rescue, IEEE Intell. Syst. **15**(2), 14–19 (2000)

53.242 G. Sukhatme, J.F. Montgomery, R.T. Vaughan: Experiments with cooperative aerial-ground robots. In: *Robot Teams: From Diversity to Polymorphism*, ed. by T. Balch, L.E. Parker (A K Peters, Natick 2002)

53.243 L.E. Parker, B. Kannan, F. Tang, M. Bailey: Tightly-coupled navigation assistance in heterogeneous multi-robot teams, Proc. IEEE Int. Conf. Intell. Robots Syst. (2004)

53.244 A. Howard, L.E. Parker, G.S. Sukhatme: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment, and detection, Int. J. Robotics Res. **25**, 431–447 (2006)

Avidbots Ex1016
Page 1398 of 2154

53.245 L. Chaimowicz, B. Grocholsky, J.F. Keller, V. Kumar, C.J. Taylor: Experiments in multirobot air-ground coordination, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

53.246 L.E. Parker, F. Tang: Building multi-robot coalitions through automated task solution synthesis, Proc. IEEE **94**(7), 1289–1305 (2006)

53.247 R. Zlot, A. Stentz: Market-based multirobot coordination for complex tasks, Int. J. Robotics Res. **25**(1), 73–101 (2006)

53.248 J. Jennings, C. Kirkwood-Watts: Distributed mobile robotics by the method of dynamic teams. In: *Distributed Autonomous Robotic Systems 3*, ed. by T. Lueth, R. Dillmann, P. Dario, H. Wörn (Springer, Berlin, Heidelberg 1998)

53.249 E. Pagello, A. D'Angelo, E. Menegatti: Cooperation issues and distributed sensing for multirobot systems, Proc. IEEE **94**, 1370–1383 (2006)

53.250 B. Gerkey, M.J. Matarić: A formal analysis and taxonomy of task allocation in multi-robot systems, Int. J. Robotics Res. **23**(9), 939–954 (2004)

53.251 D. Gale: *The Theory of Linear Economic Models* (McGraw-Hill, New York 1960)

53.252 E. Balas, M.W. Padberg: On the set-covering problem, Oper. Res. **20**(6), 1152–1161 (1972)

53.253 B.B. Werger, M.J. Matarić: Broadcast of local eligibility for multi-target observation. In: *Distributed Autonomous Robotic Systems 4*, ed. by L.E. Parker, G. Bekey, J. Barhen (Springer, Tokyo 2000)

53.254 R.A. Brooks: A robust layered control system for a mobile robot, IEEE J. Robotics Autom. **RA-2**(1), 14–23 (1986)

53.255 S. Botelho, R. Alami: M+: A scheme for multirobot cooperation through negotiated task allocation and achievement, Proc. IEEE Int. Conf. Robotics Autom. (1999) pp. 1234–1239

53.256 R.G. Smith: The contract net protocol: High-level communication and control in a distributed problem solver, IEEE Trans. Comput. **C-29**(12), 1104–1113 (1980)

53.257 B. Dias, R. Zlot, N. Kalra, A. Stentz: Market-based multirobot coordination: A survey and analysis, Proc. IEEE **94**(7), 1257–1270 (2006)

53.258 B.P. Gerkey, M.J. Matarić: Sold! Auction methods for multi-robot coordination, IEEE Trans. Robotics Autom. **18**(5), 758–768 (2002)

53.259 H. Kose, U. Tatlidede, C. Mericli, K. Kaplan, H.L. Akin: Q-learning based market-driven multi-agent collaboration in robot soccer, Proc. Turk. Symp. Artif. Intell. Neural Netw. (2004) pp. 219–228

53.260 D. Vail, M. Veloso: Multi-robot dynamic role assignment and coordination through shared potential fields, multi-robot systems: From swarms to intelligent automata, Proc. Int. Workshop Multi-Robot Syst. (2003) pp. 87–98

53.261 M. Lagoudakis, E. Markakis, D. Kempe, P. Keshinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, S. Jain: *Auction-based multi-robot routing, Robotics: Science and Systems I* (MIT Press, Cambridge 2005)

53.262 G. Rabideau, T. Estlin, S. Schien, A. Barrett: A comparison of coordinated planning methods for cooperating rovers, Proc. AIAA Space Technol. Conf. (1999)

53.263 R. Zlot, A. Stentz, M.B. Dias, S. Thayer: Multi-robot exploration controlled by a market economy, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 3016–3023

53.264 J. Guerrero, G. Oliver: Multi-robot task allocation strategies using auction-like mechanisms, Proc. 6th Congr. Catalan Assoc. Artif. Intell. (2003) pp. 111–122

53.265 N. Kalra, D. Ferguson, A. Stentz: Hoplites: A market-based framework for planned tight coordination in multirobot teams, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

53.266 L. Lin, Z. Zheng: Combinatorial bids based multi-robot task allocation method, Proc. IEEE Int. Conf. Robotics. Autom. (ICRA) (2005) pp. 1145–1150

53.267 C.-H. Fua, S.S. Ge: COBOS: Cooperative backoff adaptive scheme for multirobot task allocation, IEEE Trans. Robotics **21**(6), 1168–1178 (2005)

53.268 E.G. Jones, B. Browning, M.B. Dias, B. Argall, M. Veloso, A. Stentz: Dynamically formed heterogeneous robot teams performing tightly-coupled tasks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 570–575

53.269 Y. Zhang, L.E. Parker: IQ-ASyMTRe: Forming executable coalitions for tightly coupled multirobot tasks, IEEE Trans. Robotics **29**(2), 400–416 (2012)

53.270 L. Vig, J.A. Adams: Multi-robot coalition formation, IEEE Trans. Robotics **22**(4), 637–649 (2006)

53.271 M. Bowling, M. Veloso: Simultaneous adversarial multi-robot learning, Proc. Int. Joint Conf. Artif. Intell. (2003)

53.272 F. Fernandez, L.E. Parker: A reinforcement learning algorithm in cooperative multi-robot domains, J. Intell. Robots Syst. **43**, 161–174 (2005)

53.273 C.F. Touzet: Robot awareness in cooperative mobile robot learning, Auton. Robots **2**, 1–13 (2000)

53.274 R. Steeb, S. Cammarata, F. Hayes-Roth, P. Thorndyke, R. Wesson: *Distributed Intelligence for Air Fleet Control*, Rand Corp. Tech. Rep. R-2728-AFPA (1981)

53.275 M. Benda, V. Jagannathan, R. Dodhiawalla: On Optimal Cooperation of Knowledge Sources, Boeing AI Center Tech. Rep. BCS-G2010-28 (1985)

53.276 T. Haynes, S. Sen: Evolving behavioral strategies in predators and prey. In: *Adaptation and Learning in Multi-Agent Systems*, ed. by G. Weiss, S. Sen (Springer, Berlin, Heidelberg 1986) pp. 113–126

53.277 S. Mahadevan, J. Connell: Automatic programming of behavior-based robots using reinforcement learning, Proc. AAAI (1991) pp. 8–14

53.278 S. Marsella, J. Adibi, Y. Al-Onaizan, G. Kaminka, I. Muslea, M. Tambe: On being a teammate: Experiences acquired in the design of RoboCup teams, Proc. 3rd Annu. Conf. Auton. Agents (1999) pp. 221–227

53.279 J. Pugh, A. Martinoli: Multi-robot learning with particle swarm optimization, Proc. 5th Int. Jt.

Avidbots Ex1016
Page 1399 of 2154

Conf. Auton. Agents Multiagent Syst. (2006) pp. 441–448
53.280 P. Stone, M. Veloso: Multiagent systems: A survey from a machine learning perspective, Auton. Robots **8**(3), 345–383 (2000)
53.281 M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, K. Hosoda: Coordination of multiple behaviors acquired by a vision-based reinforcement learning, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robots Syst. (1994) pp. 917–924
53.282 M. Kubo, Y. Kakazu: Learning coordinated motions in a competition for food between ant colonies, Proc. 3rd Int. Conf. Simul. Adapt. Behav. (1994) pp. 487–492
53.283 R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert: Multi-robot cooperation in the MARTHA project, IEEE Robotics Autom. Mag. **5**(1), 36–47 (1998)
53.284 A. Stroupe, A. Okon, M. Robinson, T. Huntsberger, H. Aghazarian, E. Baumgartner: Sustainable cooperative robotic technologies for human and robotic outpost infrastructure construction and maintenance, Auton. Robots **20**(2), 113–123 (2006)
53.285 H.R. Everett, R.T. Laird, D.M. Carroll, G.A. Gilbreath, T.A. Heath-Pastore, R.S. Inderieden, T. Tran, K.J. Grant, D.M. Jaffee: *Multiple resource host architecture (MRHA) for the mobile detection assessment response system* (MDARS), SPAWAR Systems Technical Documen 3026, Revision A (2000)
53.286 Y. Guo, L.E. Parker, R. Madhavan: Towards collaborative robots for infrastructure security applications, Proc. Int. Symp. Collab. Technol. Syst. (2004) pp. 235–240
53.287 C. Hazard, P.R. Wurman, R. D'Andrea: Alphabet Soup: A testbed for studying resource allocation in multi-vehicle systems, Proc. AAAI Workshop Auction Mech. Robot Coord. (2006) pp. 23–30
53.288 D. Nardi, A. Farinelli, L. Iocchi: Multirobot systems: A classification focused on coordination, IEEE Trans. Syst. Man Cybern. B **34**(5), 2015–2028 (2004)
53.289 K. Passino: Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Syst. Mag. **22**(3), 52–67 (2002)
53.290 M. Schneider-Fontan, M. Matarić: Territorial multi-robot task division, IEEE Trans. Robotics Autom. **15**(5), 815–822 (1998)
53.291 I. Wagner, M. Lindenbaum, A.M. Bruckstein: Mac vs. PC – Determinism and randomness as complementary approaches to robotic exploration of continuous unknown domains, Int. J. Robotics Res. **19**(1), 12–31 (2000)
53.292 K. Sugawara, M. Sano: Cooperative behavior of interacting simple robots in a clockface arranged foraging field. In: *Distributed Autonomous Robotic Systems*, ed. by H. Asama, T. Arai, T. Fukuda, T. Hasegawa (Springer, Berlin, Heidelberg 2002)
53.293 P. Rybski, S. Stoeter, C. Wyman, M. Gini: A cooperative multi-robot approach to the mapping and exploration of Mars, Proc. AAAI/IAAI (1997)
53.294 S. Sun, D. Lee, K. Sim: Artificial immune-based swarm behaviors of distributed autonomous robotic systems, Proc. IEEE Int. Conf. Robotics Autom. (2001) pp. 3993–3998
53.295 A.I. Mourikis, S.I. Roumeliotis: Optimal sensor scheduling for resource-constrained localization of mobile robot formations, IEEE Trans. Robotics **22**(5), 917–931 (2006)
53.296 S. Kloder, S. Hutchinson: Path planning for permutation-invariant multirobot formations, IEEE Trans. Robotics **22**(4), 650–665 (2006)
53.297 C.W. Reynolds: Flocks, herds and schools: A distributed behavioral model, ACM SIGGRAPH Comput. Gr. **21**, 25–34 (1987)
53.298 T. Balch, R. Arkin: Behavior-based formation control for multi-robot teams, IEEE Trans. Robotics Autom. **14**(6), 926–939 (1998)
53.299 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Autom. Control **48**(6), 988–1001 (2002)
53.300 C. Belta, V. Kumar: Abstraction and control for groups of robots, IEEE Trans. Robotics **20**(5), 865–875 (2004)
53.301 C.M. Topaz, A.L. Bertozzi: Swarming patterns in two-dimensional kinematic model for biological groups, SIAM J. Appl. Math. **65**(1), 152–174 (2004)
53.302 J.A. Fax, R.M. Murray: Information flow and cooperative control of vehicle formations, IEEE Trans. Autom. Control **49**(9), 1465–1476 (2004)
53.303 J.A. Marshall, M.E. Broucke, B.R. Francis: Formations of vehicles in cyclic pursuit, IEEE Trans. Autom. Control **49**(11), 1963–1974 (2004)
53.304 S.S. Ge, C.-H. Fua: Queues and artificial potential trenches for multirobot formations, IEEE Trans. Robotics **21**(4), 646–656 (2005)
53.305 P. Tabuada, G. Pappas, P. Lima: Motion feasibility of multi-agent formations, IEEE Trans. Robotics **21**(3), 387–392 (2005)
53.306 G. Antonelli, S. Chiaverini: Kinematic control of platoons of autonomous vehicles, IEEE Trans. Robotics **22**(6), 1285–1292 (2006)
53.307 J. Fredslund, M.J. Matarić: A general algorithm for robot formations using local sensing and minimal communication, IEEE Trans. Robotics Autom. **18**(5), 837–846 (2002)
53.308 Y. Cao, W. Yu, W. Ren, G. Chen: An overview of recent progress in the study of distributed multi-agent coordination, IEEE Trans. Ind. Inf. **9**(1), 427–438 (2012)
53.309 R.M. Murray: Recent research in cooperative control of multivehicle systems, J. Dyn. Syst. Meas. Control **129**(5), 571–583 (2007)
53.310 P.J. Johnson, J.S. Bay: Distributed control of simulated autonomous mobile robot collectives in payload transportation, Auton. Robots **2**(1), 43–63 (1995)
53.311 Z. Wang, E. Nakano, T. Matsukawa: Realizing cooperative object manipulation using multiple behaviour-based robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 310–317

53.312 K. Kosuge, T. Oosumi: Decentralized control of multiple robots handling an object, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 318–323

53.313 N. Miyata, J. Ota, Y. Aiyama, J. Sasaki, T. Arai: Cooperative transport system with regrasping car-like mobile robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 1754–1761

53.314 C.R. Kube, E. Bonabeau: Cooperative transport by ants and robots, Robotics Auton. Syst. **30**(1), 85–101 (2000)

53.315 R. Groß, M. Dorigo: Towards group transport by swarms of robots, Int. J. Bio-Inspired Comput. **1**(1), 1–13 (2009)

53.316 L.E. Parker: ALLIANCE: An architecture for fault tolerant, cooperative control of heterogeneous mobile robots, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robots Syst. (1994) pp. 776–783

53.317 B. Donald, J. Jennings, D. Rus: Analyzing teams of cooperating mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (1994) pp. 1896–1903

53.318 Y. Mohan, S.G. Ponnambalam: An extensive review of research in swarm robotics, World Congr. Nat. Biol. Insp. Comput. (2009) pp. 140–145

53.319 A.J. Ijspeert, A. Martinoli, A. Billard, L.M. Gambardella: Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment, Auton. Robots **11**(2), 149–171 (2001)

53.320 A. Martinoli, K. Easton, W. Agassounon: Modeling swarm robotic systems: A case study in collaborative distributed manipulation, Int. J. Robotics Res. **23**(4–5), 415–436 (2004)

53.321 S. Berman, Q. Lindsey, M.S. Sakar, V. Kumar, S.C. Pratt: Experimental study and modeling of group retrieval in ants as an approach to collective transport in swarm robotic systems, Proc. IEEE **99**(9), 1470–1481 (2011)

53.322 J.M. Esposito: Distributed grasp synthesis for swarm manipulation with applications to autonomous tugboats, IEEE Int. Conf. Robotics Autom. (2008) pp. 1489–1494

53.323 Z. Wang, V. Kumar: Object closure and manipulation by multiple cooperating mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 394–399

53.324 Z. Wang, Y. Hirata, K. Kosuge: Control a rigid caging formation for cooperative object transportation by multiple mobile robots, Proc. of IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 1580–1585

53.325 J. Fink, N. Michael, V. Kumar: Composition of vector fields for multi-robot manipulation via caging, Robotics Sci. Syst. Conf. (2007)

53.326 F. Arrichiello, H.K. Heidarsson, S. Chiaverini, G.S. Sukhatme: Cooperative caging and transport using autonomous aquatic surface vehicles, Intell. Serv. Robotics **5**(1), 73–87 (2012)

53.327 J. Werfel: Building patterned structures with robot swarms, Proc. 19th Int. Jt. Conf. Artif. Intell. (IJCAI) (2005) pp. 1495–1502

53.328 J. Werfel, Y. Bar-Yam, D. Rus, R. Nagpal: Distributed construction by mobile robots with enhanced building blocks, IEEE Int. Conf. Robotics Autom. (2006) pp. 2787–2794

53.329 J. Werfel, R. Nagpal: Extended stigmergy in collective construction, IEEE Intell. Syst. **21**(2), 20–28 (2006)

53.330 J. Werfel, R. Nagpal: Three-dimensional construction with mobile robots and modular blocks, Int. J. Robotics Res. **27**(3/4), 463–479 (2008)

53.331 Y. Terada, S. Murata: Automatic modular assembly system and its distributed control, Int. J. Robotics Res. **27**, 445–462 (2008)

53.332 J. Wawerla, G.S. Sukhatme, M.J. Mataric: Collective construction with multiple robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 2696–2701

53.333 R.L. Stewart, R.A. Russell: Building a loose wall structure with a robotic swarm using a spatio-temporal varying template, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 712–716

53.334 R.L. Stewart, R.A. Russell: A distributed feedback mechanism to regulate wall construction by a robotic swarm, Adapt. Behav. **14**(1), 21–51 (2006)

53.335 T. Huntsberger, G. Rodriguez, P. Schenker: Robotics challenges for robotic and human Mars exploration, Proc. Robotics 2000 (2000) pp. 340–346

53.336 R.A. Brooks, P. Maes, M.J. Mataric, G. More: Lunar based construction robots, Proc. IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 389–392

53.337 C.A.C. Parker, H. Zhang, C.R. Kube: Blind bulldozing: Multiple robot nest construction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 2010–2015

53.338 S. Sen, M. Sekaran, J. Hale: Learning to coordinate without sharing information, Proc. AAAI (1994) pp. 426–431

53.339 B. Tung, L. Kleinrock: Distributed control methods, Proc. 2nd Int. Symp. High Perform. Distrib. Comput. (1993) pp. 206–215

53.340 Z.-D. Wang, E. Nakano, T. Matsukawa: Cooperating multiple behavior-based robots for object manipulation, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1524–1531

53.341 D. Rus, B. Donald, J. Jennings: Moving furniture with teams of autonomous robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 235–242

53.342 F. Hara, Y. Yasui, T. Aritake: A kinematic analysis of locomotive cooperation for two mobile robots along a general wavy road, Proc. IEEE Int. Conf. Robotics Autom. (1995) pp. 1197–1204

53.343 J. Sasaki, J. Ota, E. Yoshida, D. Kurabayashi, T. Arai: Cooperating grasping of a large object by multiple mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (1995) pp. 1205–1210

53.344 C. Jones, M.J. Matarić: Automatic synthesis of communication-based coordinated multi-robot systems, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 381–387

Avidbots Ex1016
Page 1401 of 2154

53.345 L.E. Parker: Cooperative robotics for multi-target observation, Intell. Autom. Soft Comput. **5**(1), 5–19 (1999)

53.346 A.W. Stroupe, M.C. Martin, T. Balch: Distributed sensor fusion for object position estimation by multi-robot systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 1092–1098

53.347 S. Luke, K. Sullivan, L. Panait, G. Balan: Tunably decentralized algorithms for cooperative target observation, Proc. 4th Int. Jt. Conf. Auton. Agents Multiagent Syst. (2005) pp. 911–917

53.348 S.M. LaValle, H.H. Gonzalez-Banos, C. Becker, J.-C. Latombe: Motion strategies for maintaining visibility of a moving target, Proc. IEEE Int. Conf. Robotics Autom. (1997) pp. 731–736

53.349 T.H. Chung, J.W. Burdick, R.M. Murray: A decentralized motion coordination strategy for dynamic target tracking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2416–2422

53.350 A. Kolling, S. Carpin: Multirobot cooperation for surveillance of multiple moving targets – A new behavioral approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1311–1316

53.351 B. Jung, G. Sukhatme: Tracking targets using multiple mobile robots: The effect of environment occlusion, Auton. Robots **13**(3), 191–205 (2002)

53.352 Z. Tang, U. Ozguner: Motion planning for multitarget surveillance with mobile sensor agents, IEEE Trans. Robotics **21**(5), 898–908 (2005)

53.353 M.A. Vieira, R. Govindan, G.S. Sukhatme: Scalable and practical pursuit-evasion with networked robots, Intell. Serv. Robotics **2**(4), 247–263 (2009)

53.354 L.E. Parker: Path planning and motion coordination in multiple mobile robot teams. In: *Encyclopedia of Complexity and System Science*, ed. by R.A. Meyers (Springer, Berlin, Heidelberg 2009)

53.355 D. Grossman: Traffic control of multiple robot vehicles, IEEE J. Robotics Autom. **4**, 491–497 (1988)

53.356 P. Caloud, W. Choi, J.-C. Latombe, C. Le Pape, M. Yim: Indoor automation with many mobile robots, IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 67–72

53.357 H. Asama, K. Ozaki, H. Itakura, A. Matsumoto, Y. Ishida, I. Endo: Collision avoidance among multiple mobile robots based on rules and communication, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1991) pp. 1215–1220

53.358 S. Yuta, S. Premvuti: Coordinating autonomous and centralized decision making to achieve cooperative behaviors between multiple mobile robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (1992) pp. 1566–1574

53.359 J. Wang: Fully distributed traffic control strategies for many-AGV systems, IEEE Int. Workshop Intell. Robots Syst. (1991) pp. 1199–1204

53.360 J. Wang, G. Beni: Distributed computing problems in cellular robotic systems, Proc. IEEE Int. Workshop Intell. Robots Syst. (1990) pp. 819–826

53.361 P. Svestka, M. Overmars: Coordinated path planning for multiple robots, Robotics Auton. Syst. **23**, 125–152 (1998)

53.362 M. Peasgood, C. Clark, J. McPhee: A complete and scalable strategy for coordinating multiple robots within roadmaps, IEEE Trans. Robotics **24**(2), 283–292 (2008)

53.363 M. Erdmann, T. Lozano-Perez: On multiple moving objects, Algorithmica **2**, 477–521 (1987)

53.364 C. Ferrari, E. Pagello, J. Ota, T. Arai: Multirobot motion coordination in space and time, Robotics Auton. Syst. **25**, 219–229 (1998)

53.365 M. Bennewitz, W. Burgard, S. Thrun: Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobiel robots, Robotics Auton. Syst. **41**(2), 89–99 (2002)

53.366 H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, H. Matasubara: RoboCup: A challenge problem of AI, AI Magazine **18**(1), 73–86 (1997)

53.367 H. Kitano, S. Tadokoro: RoboCup rescue: A grand challenge for multiagent and intelligent systems, AI Magazine **22**(1), 39–52 (2001)

53.368 B. Browning, J. Bruce, M. Bowling, M. Veloso: STP: Skills, tactics and plays for multi-robot control in adversarial environments, IEEE J. Control Syst. Eng. **219**, 33–52 (2005)

53.369 M. Veloso, P. Stone, K. Han: The CMUnited-97 robotic soccer team: Perception and multiagent control, Robotics Auton. Syst. **29**(2/3), 133–143 (1999)

53.370 T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, B. Nebel: CS Freiburg: Coordinating robots for successful soccer playing, IEEE Trans. Robotics Autom. **5**(18), 685–699 (2002)

53.371 P. Stone, M. Veloso: Task decomposition, dynamic role assignemnt, and low-bandwidth communicaiton for real-time strategic teamwork, Artif. Intell. **110**(2), 241–273 (1999)

53.372 C. Candea, H.S. Hu, L. Iocchi, D. Nardi, M. Piaggio: Coordination in multi-agent Robocup teams, Robotics Auton. Syst. **36**(2), 67–86 (2001)

53.373 L.E. Parker: Current state of the art in distributed autonomous mobile robotics, Distrib. Auton. Robotic Syst. **4**, 3–12 (2000)

53.374 K.R. Baghaei, A. Agah: Task allocation and communication methodologies for multi-robot systems, Intell. Autom. Soft Comput. **9**, 217–226 (2003)

53.375 T. Balch, L.E. Parker: Guest editorial, special issue on heterogeneous multi-robot systems, Auton. Robots **8**(3), 207–208 (2000)

53.376 M. Dorigo, E. Sahin: Guest editorial, special issue on swarm robotics, Auton. Robots **17**(2/3), 111–113 (2004)

53.377 M. Veloso, D. Nardi: Special issue on multirobot systems, Proc. IEEE **94**, 1253–1256 (2006)

53.378 T. Balch: Taxonomies of multi-robot task and reward. In: *Robot Teams: From Diversity to Polymorphism*, ed. by T. Balch, L.E. Parker (A K Peters, Natick 2002)

53.379 L.E. Parker, G. Bekey, J. Barhen (Eds.): *Distributed Autonomous Robotic Systems 4* (Springer, Berlin, Heidelberg 2000)

Avidbots Ex1016
Page 1402 of 2154

53.380 H. Asama, T. Arai, T. Fukuda, T. Hasegawa (Eds.): *Distributed Autonomous Robotic Systems 5* (Springer, Berlin, Heidelberg 2002)

53.381 R. Alami, R. Chatila, H. Asama (Eds.): *Distributed Autonomous Robotic Systems 6* (Springer, Berlin, Heidelberg 2006)

53.382 M. Gini, R. Voyles (Eds.): *Distributed Autonomous Robotic Systems 7* (Springer, Berlin, Heidelberg 2006)

53.383 H. Asama, H. Kurokawa, J. Ota, K. Sekiyama: *Distributed Autonomous Robotic Systems 8* (Springer, Berlin, Heidelberg 2009)

53.384 A. Martinoli, F. Mondada, N. Correll, G. Mermoud, M. Egerstedt, M.A. Hsieh, L.E. Parker, K. Stoy (Eds.): *Distributed Autonomous Robotic Systems*, Springer Tracts in Advanced Robotics, Vol. 83 (Springer, Berlin, Heidelberg 2013)

53.385 A. Schultz, L.E. Parker (Eds.): *Multi-Robot Systems: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2002)

53.386 A. Schultz, L.E. Parker, F. Schneider (Eds.): *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2003)

53.387 E. Sahin, W.M. Spears (Eds.): *Swarm Robotics: SAB 2004 Int. Workshop* (Springer, Berlin, Heidelberg 2004)

53.388 L.E. Parker, F. Schneider, A. Schultz (Eds.): *Multi-Robot Systems Volume III: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2005)

53.389 T. Balch, L.E. Parker (Eds.): *Robot Teams: From Polymorphism to Diversity* (A K Peters, Natick 2002)

53.390 F. Bullo, J. Cortés, S. Martinez: *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms* (Princeton University Press, Princeton 2009)

53.391 S. Kernbach (Ed.): *Handbook of Collective Robotics: Fundamentals and Challenges* (CRC, Boca Raton 2013)

Avidbots Ex1016
Page 1403 of 2154

Multimedia Contents



Part F Robots at Work

Ed. by Alex Zelinsky

54 Industrial Robotics
Martin Hägele, Stuttgart, Germany
Klas Nilsson, Lund, Sweden
J. Norberto Pires, Coimbra, Portugal
Rainer Bischoff, Augsburg, Germany

55 Space Robotics
Kazuya Yoshida, Sendai, Japan
Brian Wilcox, Pasadena, USA
Gerd Hirzinger, Wessling, Germany
Roberto Lampariello, Wessling, Germany

56 Robotics in Agriculture and Forestry
Marcel Bergerman, Pittsburgh, USA
John Billingsley, Toowoomba, Australia
John Reid, Moline, USA
Eldert van Henten, Wageningen, The Netherlands

57 Robotics in Construction
Kamel S. Saidi, Gaitherbsurg, USA
Thomas Bock, Munich, Germany
Christos Georgoulas, Munich, Germany

58 Robotics in Hazardous Applications
James Trevelyan, Crawley, Australia
William R. Hamel, Knoxville, USA
Sung-Chul Kang, Seoul, Korea

59 Robotics in Mining
Joshua A. Marshall, Kingston, Canada
Adrian Bonchis, Pullenvale, Australia
Eduardo Nebot, Sydney, Australia
Steven Scheding, Sydney, Australia

60 Disaster Robotics
Robin R. Murphy, College Station, USA
Satoshi Tadokoro, Sendai, Japan
Alexander Kleiner, Linköping, Sweden

61 Robot Surveillance and Security
Wendell H. Chun, Denver, USA
Nikolaos Papanikolopoulos, Minneapolis, USA

62 Intelligent Vehicles
Alberto Broggi, Parma, Italy
Alex Zelinsky, Canberra, Australia
Ümit Özgüner, Columbus, USA
Christian Laugier, Saint Ismier, France

63 Medical Robotics and Computer-Integrated Surgery
Russell H. Taylor, Baltimore, USA
Arianna Menciassi, Pisa, Italy
Gabor Fichtinger, Kingston, Canada
Paolo Fiorini, Verona, Italy
Paolo Dario, Pisa, Italy

64 Rehabilitation and Health Care Robotics
H.F. Machiel Van der Loos, Vancouver, Canada
David J. Reinkensmeyer, Irvine, USA
Eugenio Guglielmelli, Rome, Italy

65 Domestic Robotics
Erwin Prassler, Sankt Augustin, Germany
Mario E. Munich, Pasadena, USA
Paolo Pirjanian, Bedford, USA
Kazuhiro Kosuge, Sendai, Japan

66 Robotics Competitions and Challenges
Daniele Nardi, Rome, Italy
Jonathan Roberts, Brisbane, Australia
Manuela Veloso, Pittsburgh, USA
Luke Fletcher, Brisbane, Australia

Avidbots Ex1016
Page 1404 of 2154

Part F, Robots at Work, covers the advances in technology that are concerned with the growing area of robot applications, ranging from factory robotics, through a diverse array of industry applications such mining, agriculture, construction to health care and domestic robotics. The future vision for robotics is for the pervasive application of robots. The robots of the future will perform all the dangerous, dirty, and dreary (DDD) tasks. Joe Engelberger, the pioneer of the robotics industry, wrote in his 1989 book *Robotics in Service* that the inspiration to write the book came as a reaction to a forecast study of robot applications, which predicted that in 1995 applications of robotics outside factories (the traditional domain of industrial robots) would account for less than 1% of total sales. Engelberger believed that this forecast was wrong, and he instead predicted that the non-industrial class of robot applications would become the largest class of robot applications. Engelberger's prediction has yet to come to pass. However, he did correctly foresee the growth in non-traditional applications of robots. Previous parts of this Handbook show the great strides that robotics technology has made in the past 50 years. The technology has reached a level of maturity such that robots are now marching from the factories into field and service applications.

The topics in Part F cover the essentials of what is required to create robots that can operate in all environments and perform meaningful work. Part F describes fit-for-purpose robots and includes hardware design, control (of locomotion, manipulation, and interaction), perception, and user interfaces. The economic/social drivers for the particular applications are also discussed. Part F builds on all of the previous parts of the Handbook. Robotics Foundations (Part A) and Design (Part B) are essential for providing the basic mechanisms and control structures for any robot that is targeted for applications work. Sensing and Perception (Part C) and Manipulation and Interfaces (Part D) are critical capabilities for robots that need to interact with changing environments and perform manipulation tasks under human supervision. To fully utilize the potential of a working robot requires mobility and the ability to cooperate with other robotics, the basic technologies described in Moving in the Environment (Part E) are essential.

Chapter 54, Industrial Robotics, gives a history and description of typical industrial robot applications. Most robots today can trace their origin to early industrial robot designs. All the important foundations for robot control were initially developed with industrial applications in mind. The chapter describes how robot with different mechanisms are designed to fit different applications.

Chapter 55, Space Robots and Systems, deals with robotics technology for space applications. It covers two classes of robotics: orbital systems and operation of planetary surfaces. Any unmanned spacecraft is a robotic spacecraft. However, space robots are more capable devices that can facilitate manipulation, assembly, assisting astronauts, and extending exploration to remote planets as surrogates for human explorers. This chapter covers the key issues in space robots and systems: manipulation, mobility, sensing and perception, tele-operation and autonomy, and dealing with extreme environments. A historical overview and a review of the most recent technical advances for space robots is provided. An extensive description of the mathematics for the control of robotic devices in microgravity environments is also provided.

Chapter 56, Robotics in Agriculture and Forestry, describes how robotics has made a impact in the agricultural and forestry industries. Agriculture is an industry that is driven by productivity. Growing human population drives ever-increasing demands for greater food production. Robotics is a productivity technology that can be game changer in agriculture. The chapter focuses on case studies where robotics has made a marked difference to agriculture and forestry. The case studies cover the use of robotics in field crops, weeding, seeding, irrigation, orchard crops – fruit and vegetables, forestry, and livestock applications – breeding, harvesting, slaughter and processing. The case studies follow the progression of robotics technology from sensing through mobility to manipulation. The chapter emphasizes the agricultural field as a fruitful source for robotics applications, that are sufficiently demanding to require the development of new techniques, methods and technologies.

Chapter 57, Robotics in Construction, describes the construction automation concepts that have been developed and presents examples of construction robots that are in use and are in various stages of development. The chapter gives an overview of the construction industry and discusses the concept of automation versus robotics. The chapter discusses robotics in applications that are on construction sites. The chapter deals with the growing number robotics and automation application of offsite construction tasks such as prefabrication. Unsolved technical challenges including interoperability, connection systems, tolerances, power, and communications are discussed.

Chapter 58, Robotics in Hazardous Applications, discusses robot technologies for dealing with difficult and dangerous environments. The technology solutions that are adopted depend on the nature and magnitude of

the hazards. When the magnitudes of hazards reach the point that human exposure represents a direct threat to life or has long-term health consequences, e.g., nuclear radiation, some form of remote operation must be used. This chapter describes robotics technologies for dealing with eradicating landmines, and hazardous materials handling, such as explosives disposal, and handling of dangerous biological and nuclear material. Enabling technologies to support demining and hazardous material handling are discussed, including mobile platforms, manipulator design, teleoperation and control, and the supply of energy and communication signals through reliable and robust tethers. The chapter ends with a discussion of the advances that are required to progress the field. Progress is needed in mechatronics design, sensing, machine intelligence, and fully understanding the application and its associated economic factors.

Chapter 59, Mining Robotics, discusses the application of robotics to the mining industry. Mining remains physically demanding and hazardous work. The scope for the application of robotics is great since mining requires the handling of enormous quantities of material in a cost-effective and safe manner. High operational costs, the need for greater productivity, and improved health and safety outcomes are powerful drivers for robotics. Robotics and automation is providing the next step change in productivity and safety. This chapter reviews the modern mining practice and the technology drivers. It describes the enabling robot technologies associated with surface mining and underground mining. The chapter covers the technologies associated with digging, drilling, explosives handling, haulage, truck fleet management, autonomous mapping of mines and robotic rescue for recover in mining accidents. The chapter considers the challenges associated with mining in extreme environments such as undersea extraction of resources and the long-term possibilities of interplanetary exploration and mining.

Chapter 60, Disaster Robotics, describes the state of the art of an emerging applications field. In recent times robots have served as extensions of responders to a disaster, providing real-time video and other sensory data about the situation. The technology is still emerging and is beginning to be adopted by the international emergency response community. Robots are being used by fire departments and for bomb disposal. This chapter describes the basic nature of disasters, the impact on robot design, the types of robots actually used in disasters such as the Fukushima Nuclear disaster, promising designs, future concepts and benchmarks for evaluation of disaster robotics. The chapter discusses the fundamental challenges facing disaster robotics, including the key challenge of evolution from a concept to the adoption as a solution technology.

Chapter 61, Robot Surveillance and Security introduces surveillance and security robots for military and civil applications. The chapter describes the applications environment that covers ground, aerial and maritime domains for a wide range of surveillance and security robots. A systems overview is provided of the mobility component, sensor payload, communication system, and operator control interface, together with a description of the enabling technologies. The chapter concludes with a review of current research topics and discusses the future direction of robotic surveillance and security.

Chapter 62, Intelligent Vehicles, describes the emerging robotics application field of intelligent vehicles: motor vehicles that have autonomous functions and capabilities. The chapter describes why the development of intelligent vehicles is important and gives a brief history of the field and the potential benefits of the technology. It describes the enabling technologies for intelligent vehicles to sense vehicle, environment, and driver state, work with digital maps and satellite navigation, and communicating with intelligent transportation infrastructure. The chapter describes the challenges and solutions associated with road scene understanding. Advanced driver assistance systems which use robotics technologies to create new safety and convenience systems for motor vehicles, such as collision avoidance, lane keeping, parking assistance, as well as driver monitoring to mitigate against driver fatigue, inattention, and impairment are described.

Chapter 63, Medical Robotics and Computer-Integrated Surgery describes the development of medical robotics. In the last 20 years starting with brain surgery, orthopaedics, endoscopic surgery, and microsurgery, the field has expanded to include commercially marketed, clinically deployed systems and an active research community. This chapter provides a historical review of the field and discusses the major thrusts using examples from current and past research. Medical robots are described within the context of larger computer-integrated systems including presurgical planning, intraoperative execution, and postoperative assessment and follow-up. The basic concepts of computer-integrated surgery including the critical factors affecting the deployment and acceptance of medical robots are described. An overview of medical robot systems is provided, including remote telesurgery and robotic surgical simulators. The chapter concludes with a discussion on future research directions.

Chapter 64, Rehabilitation and Health Care Robotics, describes robotic systems that assist persons who have a disability or provide rehabilitation therapy for persons. The chapter provides a historical review of the field. Physical therapy and training robots as well as robotic aids for people with disabilities are described. Advances in smart prostheses and orthoses for rehabilitation robotics are also described. An overview of work in diagnosis and monitoring for rehabilitation is provided. The chapter provides a solid understanding of the future challenges for rehabilitation and heath care robotics.

Chapter 65, Domestic Robots, describes the technology that everyone would one day like in their home, which will vacuum the house, clean the kitchen, load the dishwasher, or polish the shoes. In spite of hundreds of millions of potential users surprisingly not many domestic robots exist, with one notable exception – the Roomba! This chapter examines what have been the success factors for commercializing domestic robots using cleaning robots as a representative example. The chapter reviews domestic floor cleaning robots, robotic pool cleaners, window cleaning robots, and robotic lawn mowing. The chapter also examines smart appliances such as sports robotics, and telepresence robots. A description of research projects for smart environments and smart homes is presented. The chapter describes the important underpinning technologies for domestic robots covering sensing, obstacle avoidance, localization, mapping and coverage.

Chapter 66, Robotics Competitions and Challenges, explores the use of competitions to accelerate robotics research and to promote education. Robot competitions have been shown to drive innovation in the robotics field. The chapter covers the two broad types of robot competition: human-inspired competitions and task based challenges. Human-inspired robot competitions, which are mostly sports contests, develop platforms that support problem solving. Task based challenges attract participants by presenting a tough new challenge for robotics technology. Three case studies of robot competitions are presented Robot Soccer, the Unmanned Aerial Vehicles (UAV) Challenge and the DARPA (Defense Advanced Research Projects Agency) Grand Challenges. The case studies describe the organizational challenges for participants, the benefits and limitations of competitions and what it takes to have a *good* robot competition.

Multimedia Contents



54. Industrial Robotics

Martin Hägele, Klas Nilsson, J. Norberto Pires, Rainer Bischoff

Much of the technology that makes robots reliable, human friendly, and adaptable for numerous applications has emerged from manufacturers of industrial robots. With an estimated installation base in 2014 of about 1.5 million units, some 171 000 new installations in that year and an annual turnover of the robotics industry estimated to be US$ 32 billion, industrial robots are by far the largest commercial application of robotics technology today.

The foundations for robot motion planning and control were initially developed with industrial applications in mind. These applications deserve special attention in order to understand the origin of robotics science and to appreciate the many unsolved problems that still prevent the wider use of robots in today's agile manufacturing environments. In this chapter, we present a brief history and descriptions of typical industrial robotics applications and at the same time we address current critical state-of-the-art technological developments. We show how robots with different mechanisms fit different applications and how applications are further enabled by latest technologies, often adopted from technological fields outside manufacturing automation.

We will first present a brief historical introduction to industrial robotics with a selection of contemporary application examples which at the same time refer to a critical key technology. Then, the basic principles that are used in industrial robotics and a review of programming methods will be outlined. We will also introduce the topic of system integration particularly from a data integration point of view. The chapter will be closed with an outlook based on a presentation of some unsolved problems that currently inhibit wider use of industrial robots.

54.1 **Industrial Robotics: The Main Driver for Robotics Research and Application** . 1386
54.2 **A Short History of Industrial Robots** 1386
54.3 **Industrial Robot Kinematics** 1392
54.4 **Typical Industrial Robot Applications** 1393
54.4.1 Handling 1393
54.4.2 Welding 1396
54.4.3 Assembly 1398
54.4.4 Painting 1402
54.4.5 Processing 1402
54.5 **Safe Human–Robot Collaboration** 1405
54.5.1 Overview of Basic Robot Safety Standards 1405
54.5.2 Types and Requirements of Human–Robot Collaboration .. 1406
54.5.3 Examples of Human–Robot Collaboration 1408
54.6 **Task Descriptions – Teaching and Programming** 1409
54.7 **System Integration** 1414
54.8 **Outlook and Long-Term Challenges** 1416
Video-References 1418
References 1418

Avidbots Ex1016
Page 1408 of 2154

54.1 Industrial Robotics: The Main Driver for Robotics Research and Application

Even though robots are considered a cornerstone of today's competitive manufacturing particularly in automobile and related component assembly, there are still challenges to solve for manufacturing to efficiently respond to changing consumer behavior and global shifts in competitiveness. Furthermore, high-growth industries (in electronics, food, logistics, and life-sciences) and emerging manufacturing processes (gluing, coating, laser-based processes, precision assembly, fiber material processing) as well as fulfilling sustainability regulations will increasingly depend on advanced robot technology [54.1]. Additionally, the range of feasible applications could significantly increase if robots were easier to install, to integrate with other manufacturing processes, and to program, particularly with adaptive sensing and automatic error recovery. Further challenges result from the integration of various types of controls (programmable logic controller (PLC), computer numerical control (CNC) sensors) with the robot controller, from close human–robot collaboration and fenceless production with both lightweight and heavy duty robots, and from an increasing need to save energy.

Design and production of industrial robots on the one hand, and the planning, integration, and operation of robot work cells on the other hand are largely independent engineering tasks. In order to be produced in sufficiently large quantities, a robot design should meet the requirements for the widest set of potential applications. As this is difficult to achieve in practice, various classes of robot designs regarding payload capacity, number of robot axes, and workspace volume have emerged for application categories such as assembly, palletizing, painting, welding, machining, and general handling tasks.

Generally, a robot workcell consists of one or more robots with controllers and so-called robot peripherals, e.g., grippers or tools, safety devices, sensors, and material transfer components for moving and presenting parts. Typically, the cost of a complete robot workcell is four to five times the cost of the robots alone; however, efforts are underway to drastically reduce these costs through use of increased robot functionality and artificial intelligence [54.2]. A robot workcell is usually the result of customized planning, integration, programming, and configuration, requiring significant engineering expertise. Standardized engineering methods, tools, and best-practice examples for specifying and designing robot workcells have become available to provide predictable performance and to secure investments [54.3].

Today's industrial robots are mainly rooted in the requirements of capital-intensive large-volume manufacturing, typically defined by the automotive, electronics, and electrical goods industries which make up 80% of all robot installations. Future industrial robots will not be a mere extrapolation of today's designs, but will rather follow new design principles addressing a much wider range of application areas and industries. At the same time, new technologies, particularly from the information technology (IT) or the consumer domain will have an increasing impact on the design, performance, use and cost of future industrial robots.

International and national standards now help to quantify robot performance and define safety precautions, geometry, and media interfaces [54.4]. Most robots operate behind secure barriers to keep people at a safe distance. Recently, improved safety standards have allowed direct human–robot collaboration, permitting robots and human factory workers to share the same workspace [54.5, 6].

54.2 A Short History of Industrial Robots

The invention of the industrial robot dates back to 1954 when inventor George Devol filed a patent on a *programmed article transfer* (Fig. 54.1). After teaming up with young engineer and entrepreneur Joseph Engelberger, the first robot company, Unimation, was founded. It put the first robot into service at a General Motors plant in 1961 for extracting parts from a die-casting machine. Most of the hydraulically actuated Unimates were sold through the following years for workpiece handling and for spot-welding of car bodies [54.7]. Soon, many other companies started to develop and manufacture industrial robots in many industrial nations; an innovation-driven industry was born [54.8]. The first International Symposium on Industrial Robotics (now ISR) took place in Chicago in 1970 and proved that robotics had become the field of activity of a vibrant research community.

The breakthrough Stanford Arm was designed as a research prototype in 1969 by Victor Scheinman (Chap. 4). The six-degree-of-freedom (6-DOF) all-

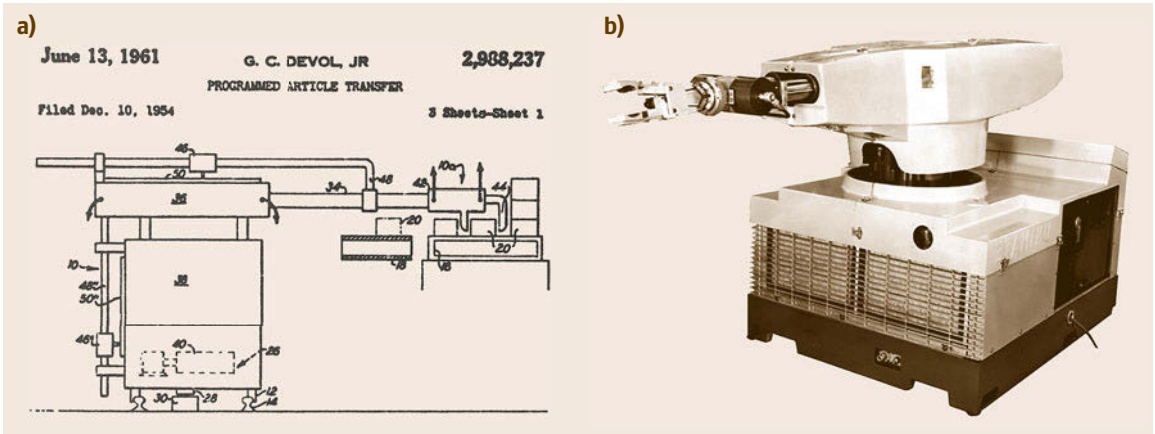


Fig.54.1a,b The invention of the industrial robot. **(a)** This patent was the start of a joint effort between G. Devol and J. Engelberger to form the first robot company, Unimation, a fusion of the terms *universal* and *automation*. The company was acquired by Westinghouse in the late 1980s and subsequently taken up by Stäubli in 1988. **(b)** The first Unimation performed a rather simple handling task in 1961 at a General Motors plant; other car manufacturers followed (courtesy of Smithsonian Institution Archives, Washington DC)

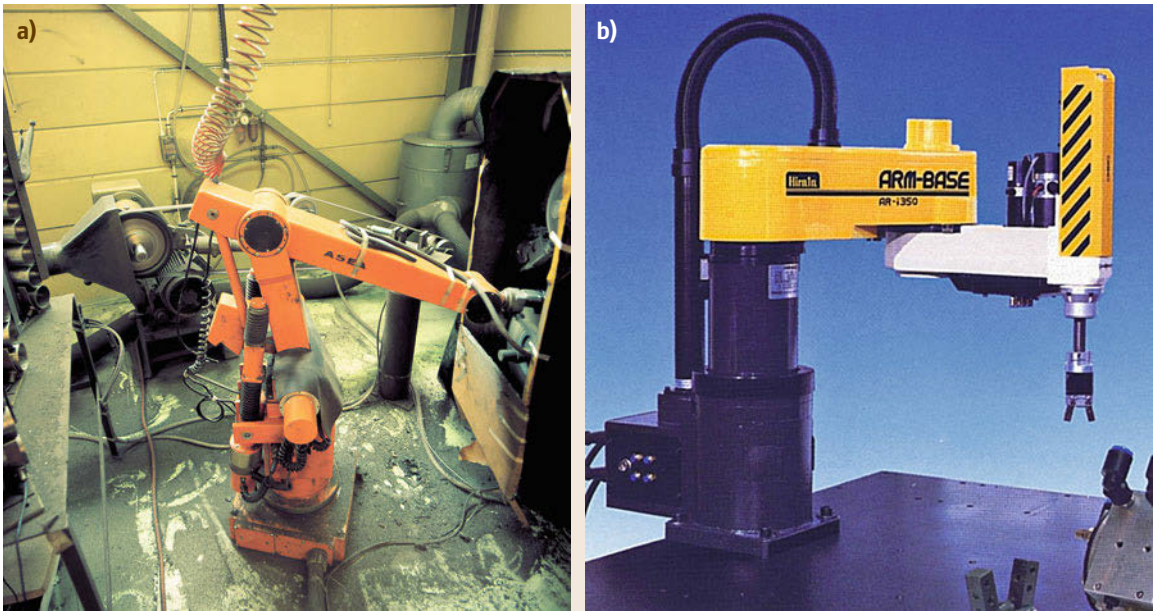


Fig.54.2a,b The all-electric **(a)** IRB-6 and **(b)** a SCARA-type kinematic. **(a)** First introduced in 1973, the IRB-6 has been a breakthrough development as it was the first serially produced robot product, which combined all-electric-drives technology and a microcomputer for motion control and programming. The robot proved very robust, and life-times of more than 25 years in harsh productions were reported (courtesy of ABB Automation). **(b)** The selective compliance assembly robot arm (SCARA) is particularly suited for assembly tasks as it combines rigidity in the vertical axis and compliance in the horizontal axis. In 1978, the first Hirata AR-300 was put together. Depicted is the successor design, the AR-i350 (courtesy of HIRATA Robotics, Mainz)

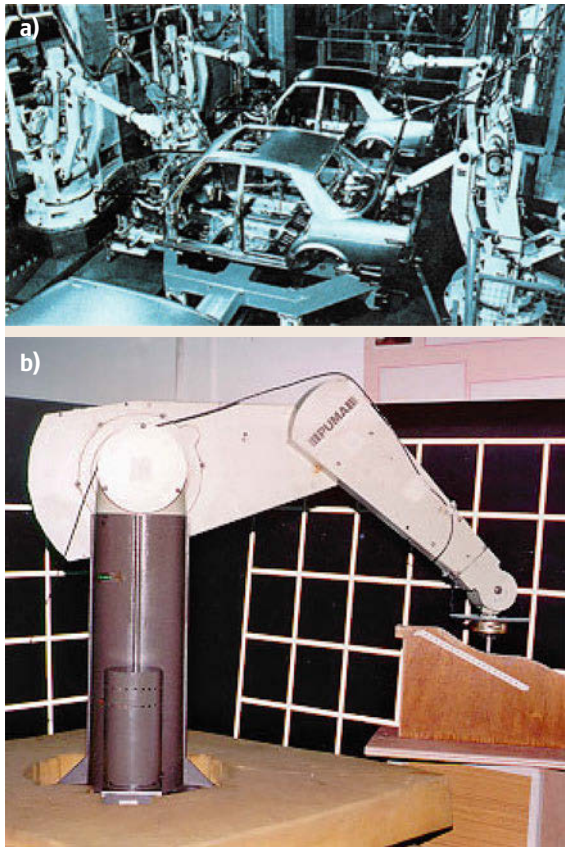


Fig.54.3a,b The KUKA IR 601/60 **(a)** and the Unimation PUMA (programmable universal machine for assembly) 560 **(b)**. **(a)** In 1978, the novel 6 DOF KUKA robot featured a parallel linkage for its second and third axes. At almost two tons of own weight, it could handle payloads of some 60 kg at maximum operating speed. The robot quickly became a workhorse for the automotive industry. **(b)** The six axis PUMA was inspired by the dexterity of a human arm. After its launch in 1979 by Unimation, it became one of the most popular arms and was used, due to its versatility and ease of use, as a reference in robotics research for many years

electric manipulator was controlled by a state-of-the-art computer of the time, a DEC PDP-6. The nonanthropomorphic kinematic configuration with one prismatic and five rotational joints was configured such that the equations for solving the robot kinematics were simple enough to speed up computations. Drives consisted of direct-current (DC) electric motors, harmonic drive and spur gear reducers, potentiometers and tachometers for position and velocity feedback [54.9]. Subsequent robot designs were strongly influenced by Scheinman's concepts (Figs. 54.2 and 54.3).

In 1973, the company ASEA (now ABB) introduced the first microcomputer-controlled all-electric industrial robot, the IRB-6, which allowed continuous path motion (CP), a precondition for many applications such as arc welding or material removal (Fig. 54.2). In the 1970s, intense diffusion of robots into car manufacturing set in mostly for (spot-)welding and handling applications (Fig. 54.3) [54.10].

In 1978, the selective compliance assembly robot arm (SCARA) was invented by *Makino* of Yamanashi University, Japan [54.11]. The ground-breaking four-axis low-cost design was perfectly suited for small parts assembly as the kinematic configuration allows fast and compliant arm motion (Fig. 54.4). Flexible assembly systems based on the SCARA robot in conjunction with compatible product designs (DFA) have contributed significantly to creating a boom in high-volume electronics production and consumer products [54.12]. Further optimization of robot dynamics and accuracy led to the first direct-drive SCARA robot, the AdeptOne in 1984 [54.13].

Requirements regarding a robot's speed, accuracy and weight have led to novel kinematic and transmission designs. An approach toward lightweight and stiff structures has been pursued since the 1980s by developing parallel kinematic machines (PKM) which connect the machine's basis with its end-effector by three to six parallel struts, see also Fig. 54.5. These so-called parallel robots (Chap. 4 and 18) are particularly suited to achieve short cycle times (e.g., for picking), precision (e.g., for material removal), or handling high workloads (Fig. 54.5) and have found their niches in advanced manufacturing [54.14]. However, workspace volumes tend to be significantly smaller than those of serial or open kinematic chain robots which are comparable in size.

Efforts of reducing mass and inertia of serial robot structures have been a primary research target, where the human arm with a weight-to-load ratio better than 1 : 1 was considered the ultimate benchmark. In 2006, robot manufacturer KUKA introduced their LBR lightweight prototype robot, a compact 7-DOF robot arm with advanced torque-control capabilities which has recently been introduced in high-performance industrial applications [54.15]. An obvious next step in approaching human dexterity is the recent introduction of two-armed robot designs with some recent developments being depicted in Fig. 54.7 [54.16]. In conjunction with a robot's capability to support safe human–robot collaboration, new manufacturing concepts can be implemented which expand capabilities, productivity, and ergonomic quality to manual workplaces [54.17].

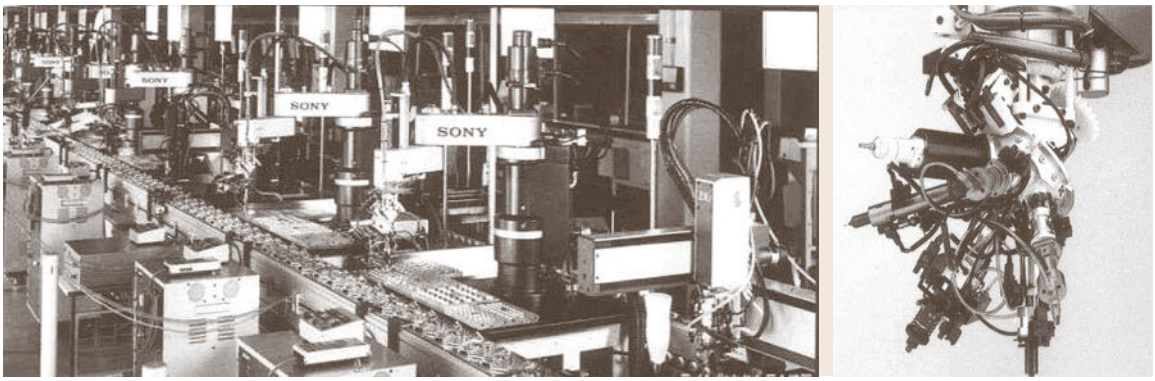


Fig. 54.4 An automated video cassette recorder (VCR) assembly line (about 1989) with SCARAs carrying a turret with multigripper tools. Typically five parts are added by one robot before the VCR is moved to the next station of the automated assembly line

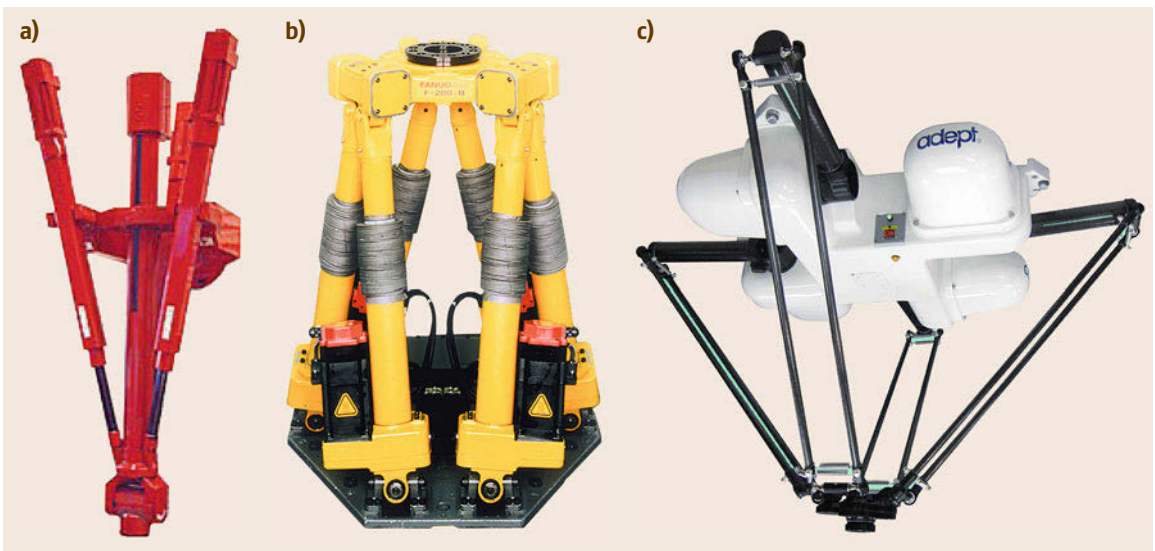


Fig.54.5a–c Parallel robots are slowly diffusing into various fields of industrial application: **(a)** the Neos Tricept 600, **(b)** Fanuc F-200iB. **(c)** Adept Quattro. **(a)** In 1992, Neos Robotics represented with their Tricept robot range a concept to combine the stiffness of machine tools with the dexterity of a robot for heavy-duty applications such as in friction stir welding (FSW) or machining of aluminum for the aircraft industry. **(b)** The Fanuc F-200iB introduced in 2002 is a 6-DOF parallel robot particularly designed for welding gun handling, deflashing, or for assembly tasks (100 kg payload, ± 0.1 mm accuracy) in automotive assembly processes; **(c)** the Adept Quattro (introduced in 2007, following the ABB FlexPicker in 1998) is suited for high-speed applications in packaging, manufacturing, assembly, and material handling. The quad dual-link arm design forms an over-determined kinematic linkage, which in the wrist is converted to the forth axis of end-effector rotation by means of an internal transmission in the wrist

In parallel to industrial robots, automated guided vehicles (AGV) have emerged. These mobile robots are used for moving workpieces or loading equipment following a predetermined or virtual path in industrial environments. Within the concept of automated flexible manufacturing systems (FMS) AGVs have become an important part of their routing flexibility. Initially, AGVs relied on prepared floors such as embedded wires, magnets, or other tags for motion guidance. Meanwhile, freely navigating AGVs along virtual trajectories are entering large-scale manufacturing and logistics. Usually, their navigation is based on laser scanners that provide an accurate two- or even three-dimensional map of the actual environment for self-localization and obstacle avoidance. Early on, combinations of AGVs and robot arms were sug-

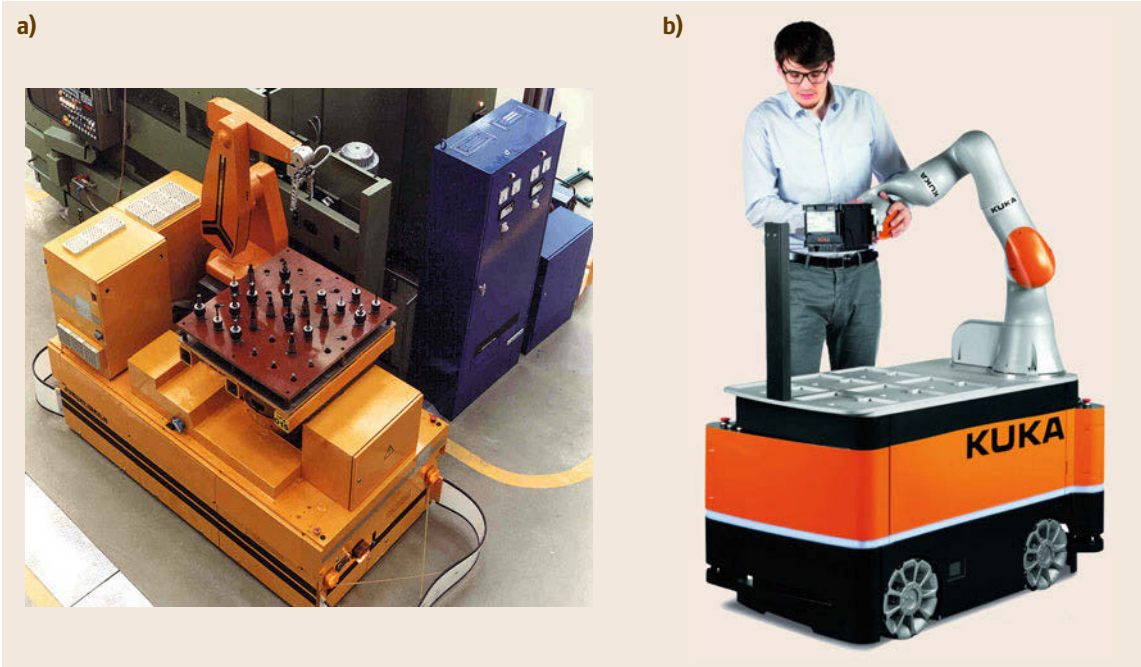


Fig.54.6a,b Mobile robots were introduced in the early 1980s for increased flexibility and reliability in factory logistics. **(a)** The MORO (1984) developed at Fraunhofer IPA was one of the first prototypes to combine a robot arm on a wire-bound mobile platform which follows a wire buried in the floor. **(b)** The KUKA omniRob features an omnidirectional platform and the LBR iiwa lightweight arm which form a highly kinematically redundant robot system (courtesy of KUKA)

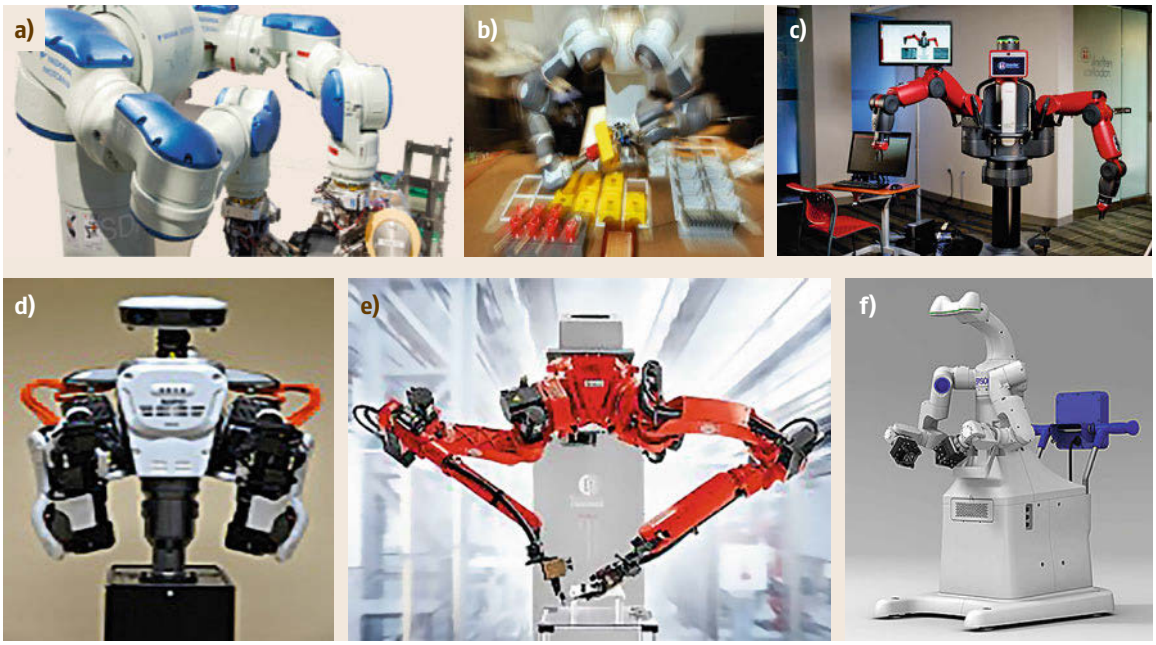


Fig.54.7a–f Examples of different designs of dual-arm robots (courtesy of **(a)** Motoman, **(b)** ABB, **(c)** Rethink Robotics, **(d)** Kawada Industries, **(e)** COMAU, **(f)** Seiko Epson)

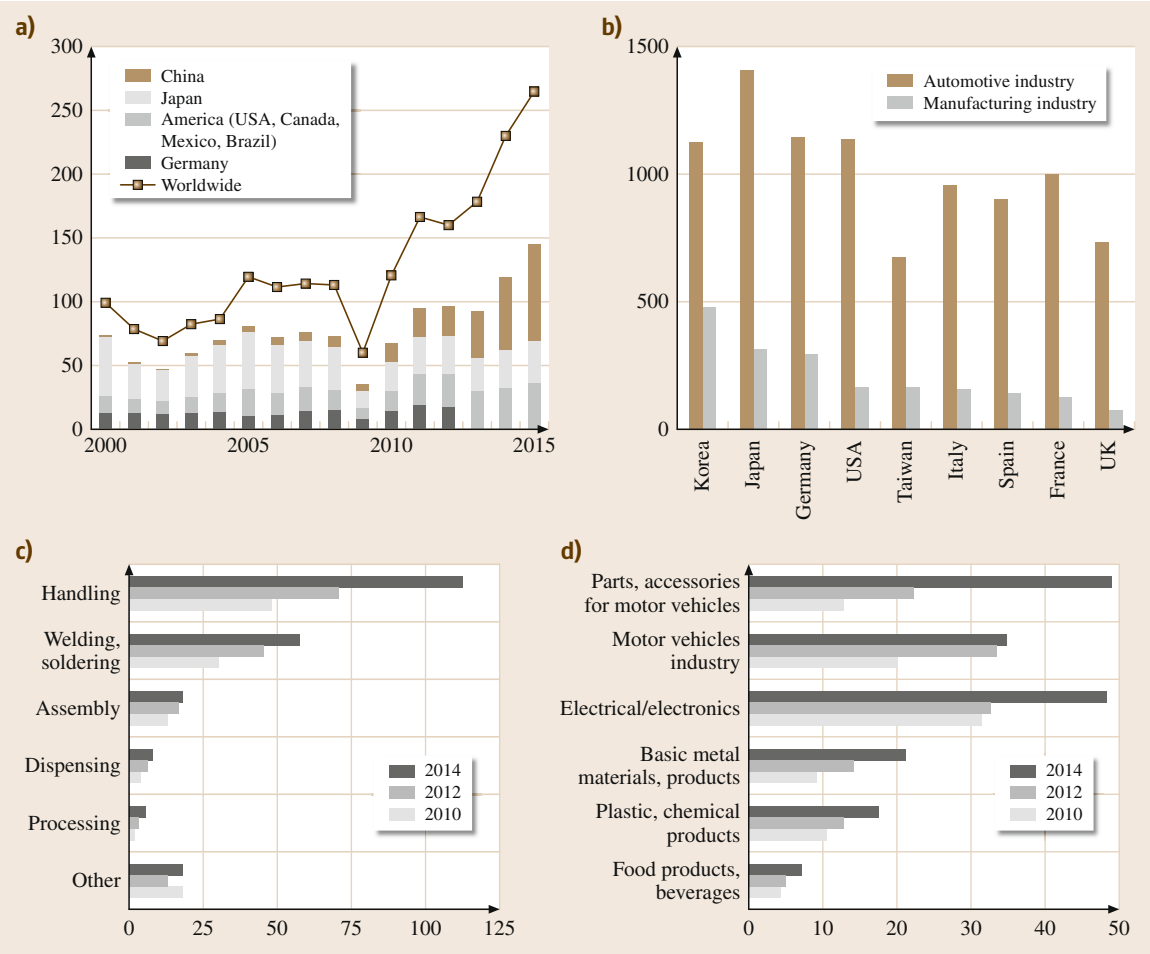


Fig.54.8a–d Statistics of worldwide industrial robotics use (after [54.1]). **(a)** Estimated annual robot installations in selected countries (1000 units, estimate for 2015), **(b)** Number of multipurpose industrial robots (all types) per 10 000 employees in the automotive and in manufacturing industries 2014. **(c)** Estimated worldwide annual shipments of industrial robots in main application areas. **(d)** Estimated worldwide annual shipments of industrial robots in main industrial branches

gested to automatically load and unload machine tools (Fig. 54.6). Safety and power supply have been an obstacle to these system's diffusion in industrial practice. Currently, first solutions for mobile manipulation appear [54.18].

The ability to use human and robot workers either interchangeably or in workspace sharing/collaboration scenarios in human workplaces motivated the design of anthropomorphic dual-arm robots (Fig. 54.7. Even though industrial acceptance initially has been low, advances in programming comfort, securing safe human–robot coexistence/collaboration and system cost have led to significant interest in using dual arms in agile manufacturing concepts, particularly in assembly and handling applications [54.19]. The dual-arm systems suggest a new way of using powerful and lean type of robot which is easy to install by the manufacturing end-user with little adaptation of manual workplaces.

Today, industrial robotics is seen as a central pillar to future manufacturing competitiveness and economic growth:

- The International Federation of Robotics (IFR) estimates that between 2000 and 2008 the robotics industry had created 8–10 million highly qualified jobs, either directly or indirectly. The prediction is that between 2012–2020 another 4 million jobs will be created in the *robot ecosystem* [54.20]. The extent of job creation by robotics has been dis-

cussed controversially. It is undisputed, however, that a wider use of robots in manufacturing is able to significantly strengthen a competitive position of a company or an industrial sector [54.21]. Economically, manufacturing productivity gains are particularly effective for economic growth. There is no sustainable product innovation without manufacturing competence which includes knowledge and practice of planning, designing, and operating advanced robotic systems [54.22] (Fig. 54.8).

- The average price for a robot in 2014 was in the order of US$ 46 800, which is about one-third of its equivalent price in 1990. At the same time, robot performance parameters such as speed, load capacity, and mean time between failures (MTBF) have dramatically improved. This means that automation has become more affordable, providing a faster return on investment [54.1].
- Traditionally, robot automation has not played a significant role in the implementation of lean manufacturing strategies. However, efforts are underway to introduce industrial robots to lean, agile manufacturing. Characteristics are robot solutions that can be flexibly added to manufacturing systems *on demand*, that are significantly less expensive on a life-cycle-costing (LCC) basis than today's systems due to reduced peripherals and systems integration (*system out of the box*) [54.23]. With robots becoming commodities in manufacturing they might be used as intuitively and naturally as a handheld power tool today. This would imply intuitive and safe human–robot collaboration and versatility due to advanced sensing, control, and embedding the robot set-up and operation in an IT infrastructure.
- Factories of the future will represent a network of self-organizing cyber physical systems (CPS). As part of this industrial internet CPSs embed computation, networking, and physical processes and can either represent manufacturing equipment such as machine tools, fixtures, trays, conveyors, tools, etc. or the workpiece which controls and memorizes its production. Robots are considered the centerpiece of future smart factories which combine manufacturing agility, profitability, human ergonomics, and minimized resource consumption [54.24].
- Robots in assembly have not reached their predicted installation potential mainly due to cheap labor cost and the lean assembly work systems which support highest flexibility and productivity at minimal waste. A reason is partly seen in the slow advances in dexterous manipulation for assembly tasks with industrial robustness. Here, torque-controlled lightweight robots [54.25] and two-armed robot systems have been proposed to imitate human ergonomics and task execution [54.26].
- New financing models such as leasing, *pay by service* will allow end-users to use robots on demand or to have manufacturing service providers to operate manufacturing lines on a *pay-on-production* basis [54.27].

Figure 54.8 depicts some key figures on the recent extent of industrial robot diffusion into manufacturing.

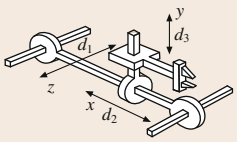
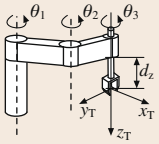
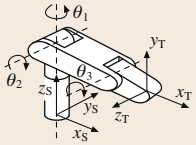
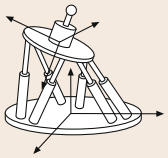
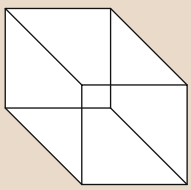
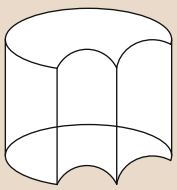
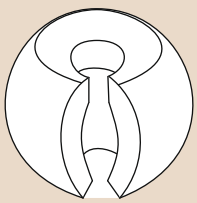
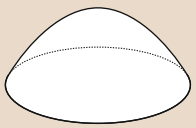
54.3 Industrial Robot Kinematics

By definition, an industrial robot is an automatically controlled, reprogrammable, multipurpose manipulator, programmable in three or more axes, which can be either fixed in place or mobile for use in industrial automation applications [54.28]. Robots can be categorized according to their number of independent kinematic axes, and their mechanical structure which affect most of the robots' kinematic properties, the computation methods used to determine joint motions, and the form and size of the robot workspace. Robot mechanical structures are composed of links that are rigid bodies connecting neighboring (prismatic, rotary, cylindrical or spherical) joints. The diagrams in Table 54.1 show several common types of robot mechanical structures. Of course, the workspace of industrial robots can be significantly expanded by placing the robot arm on an additional linear axis, sometimes reaching a length of more than 50 m, or even on mobile platforms. Furthermore, robot mechanical structures can be composed by joint modules which are connected by links to form task-specific designs.

With advances in the state of the art in motion control and computer hardware processing capabilities, computation is much less a constraint on mechanism choice than it was for early robot designers. The choice of mechanical structure of the robot depends mostly on fundamental mechanical requirements such as payload and workspace size. Considering a given level of cost, there is usually a trade-off between workspace size and stiffness. To enable the robot to reach inside or around obstacles it is clearly advantageous to use an articulated mechanical design.

Considering also the stiffness and accuracy (in a practical sense considering what is reasonable to build), the picture is more complex. Each of the first three types in Table 54.1 we refer to as serial kinematic

Table 54.1 Main categories of mechanical structures of industrial robots: Gantry is what a Cartesian coordinate robot is typically called with three prismatic joints, whose axes are coincident with a Cartesian coordinate system. The SCARA or selective compliance assembly robot arm has two parallel rotary joints to provide compliance in a selected plane. The articulated robot has three or more, typically six, rotary joints placed in series with their interconnecting links. The parallel link robot is characterized by links that form closed loop structures shown with prismatic joints, but can also have revolute joints such as the Delta robot (Fig. 54.5) (pictures courtesy of Güdel, ADEPT, ABB, PI Physik Instrumente)

Category	Gantry (or Cartesian)	SCARA	Articulate	Parallel
Robot main axes structure	 3 prismatic joints	 1 prismatic and 2(3) revolute joints	 3 revolute joints	 Typically with 3, 4 or 6 prismatic axis
Workspace shape				
Technical example				

machines (SKMs), while the last is a parallel kinematic machine (PKM). To obtain maximum stiffness, again for a certain minimum level of cost, the end-effector is better supported from different directions, and here the PKM has significant advantages. On the other hand, if high stiffness (but not low weight and high dexterity) is the main concern, a typical computerized numerical control (CNC) machine (e.g., for milling) is identical in principle to the gantry mechanism. There are also modular systems with servo-controlled actuators that can be used to build both robots with purpose-designed mechanisms.

54.4 Typical Industrial Robot Applications

Out of the many possible uses of industrial robots selected case studies on high-potential robot applications will be briefly described. Typical associated enabling technologies will be depicted.

54.4.1 Handling

Handling in robotics comprises numerous processes such as grasping, transporting, packaging, palletizing, and picking. As seen in Fig. 54.8 handling is the largest robot application field which is found in all branches of manufacturing and logistics. A central feature and major challenge in the engineering of robotic handling systems is the design of the gripper and associated grasping strategies given the physical properties of the workpiece, throughputs, and uncertainties regarding object geometry and location. Current high-potential application of robot handling systems are: tending of CNC machines for workerless shifts [54.29], palletizing, and lifting of objects for ergonomic reasons or



Fig.54.9a–d Units of sausage are cut from strings, then placed into the thermo-formed cavities before applying lidding. The coordination of the robots and the optimization of the picking frequency require a selection of the best path for each robot. Missed sausages are fed back on the conveyor for another try. The shown 4-DOF parallel robot reaches cycle times of 1–3 Hz and can move payloads of up to 8 kg (courtesy of robomotion, Germany)

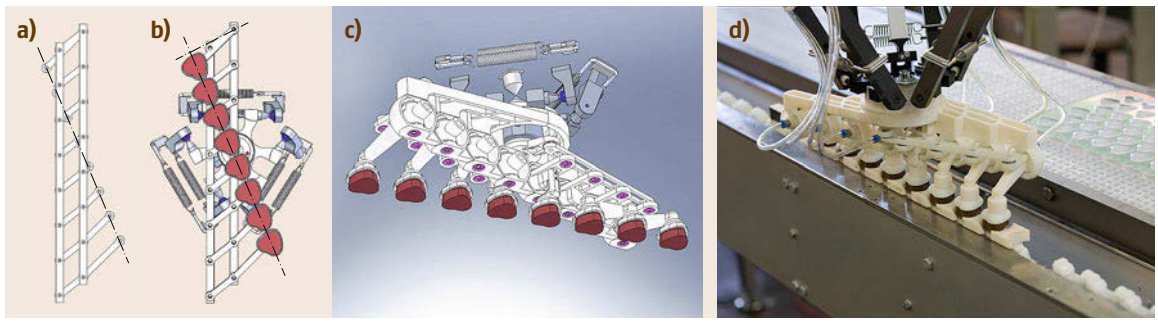


Fig.54.10a–d Lightweight customized vacuum gripper (0.75 kg mass) through additive manufacturing. Cookies are delivered continuously on a belt, grasped from the belt, and batches of eight are put on a blister matrix before final packaging. The gripper's spacing is pneumatically actuated and its rotation through the parallel robot's central rotational axis (courtesy of robomotion, Germany)

when limitations specified in load handling regulations are exceeded [54.30], for reasons of cleanliness as is typical in the food, pharmaceutical, and semiconductor industries [54.31–33], avoiding monotonous work and psychological strain, and ensure logistics quality through workpiece or object tracking [54.34]. In the following, two use-cases of material handling will be highlighted, each in a different industrial domain, and based on specific industrial robot type and enabling technologies.

Food Handling

The food sector is claimed to have significant potential for the application of robots as fundamental change to productivity, product quality, and worker ergonomics can be achieved [54.35]. In food automation, *untouched by human hand* entails critical requirements for robot automation such as the need for hygienic design, operational speed, ease of programming, and cost. In the past, these requirements had been difficult to achieve due to high throughputs, therefore requiring rapid grasps and fast robot motion, robust sensing for detecting object locations on conveyor belts. High speed at high flexibility apparently is a key in the individual handling of food objects. Therefore, fast SCARA and parallel robots have found wide acceptance in this field.

An example of a packaging line in food production is depicted in Fig. 54.9 where cut mini salamis are delivered in four streams per conveyor belt in random sequence. The positions of the sausages on the translucent belt are determined by a computer vision system. The robot picks the sausage from the measured position sequentially until the gripper holds three sausages which are then placed into cavities. With four parallel robots a maximum pick rate of 600 sausages per minute can be processed. Key of the application is its high-speed 2-D (two-dimensional) computer vision system which feeds the robot's path planning for collision-free picks at a minimum loss rate of unpicked salami [54.37].

Recent efforts have led to customized designs of gripper systems through 3-D (three-dimensional) printing (additive manufacturing) which for instance includes actuation through pneumatically driven bellows and low-wear metal joints. An example of a highly actuated gripper based on 3-D-printing for use in food handling is depicted in Fig. 54.10. Additive manufac-

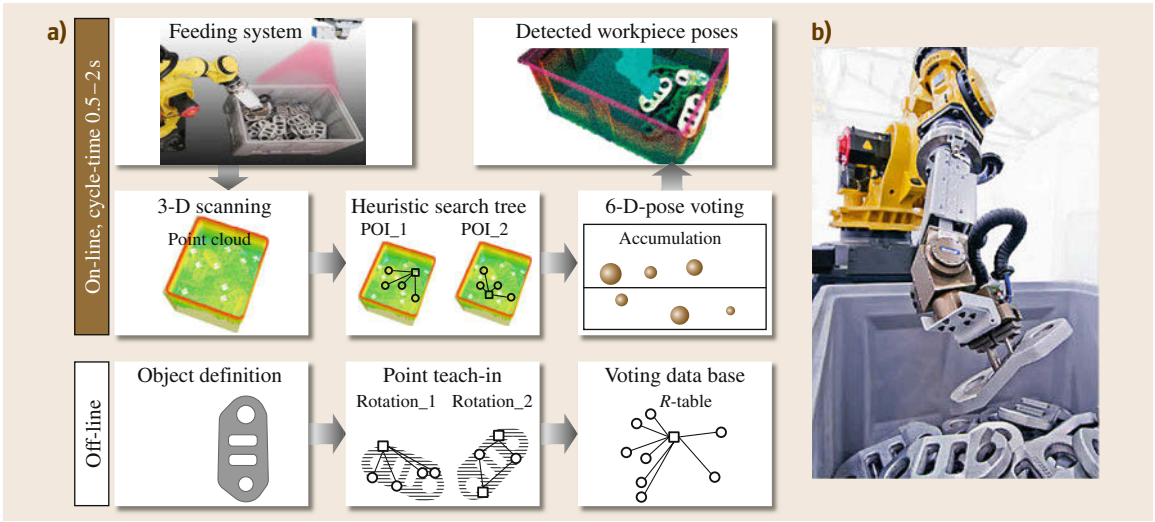


Fig.54.11a,b Procedure of a bin-picking method [54.36] **(a)** and gripper with additional degree of freedom for reaching deep into bins **(b)**. Depicted is a 2-D laser scanner on a swiveling unit for acquiring the point cloud in parallel to the robot's motion. The object detection itself consumes 0.5–2 s and is less time critical than the robot's motion and grasps

turing processes seem to be perfectly suited to achieve higher flexibility in manufacturing automation [54.38]. Numerous materials have become available for different additive manufacturing processes so that even specific manufacturing requirements can be matched. Initial doubts about gripper durability have been dispelled: lifetimes of more than 10 million load cycles for robot grippers manufactured on the basis of laser sintered polyamide have been reported.

Bin-Picking

Generally, industrial practice in robot workcell planning aims at finding a compromise between reducing the variation of the workpiece location and the cost of sensor systems to compensate for residual variation or uncertainties. Today, nearly all parts arrive at robot workcells in a repeatable manner, either being stored in special carriers or magazines, or by being transported and oriented by vibrating devices that allow the parts to settle into a predictable orientation for proper robot grasping. However, cost and flexibility requirements in manufacturing automation will result in reducing customized parts magazines to more universal carriers, containers, or conveyor belts. If randomly oriented on a conveyor belt or in a carrier, parts have to be properly identified and located so that the robot can produce an collision-free grasp.

The challenge of grasping partly or randomly ordered parts by robot has been referred to as bin-picking and has been investigated by numerous researchers since the mid-1980s [54.39]. Even though an abundance of approaches has been presented, only recently bin-picking installations have found their way into daily manufacturing in significant numbers. Bin-picking algorithms follow a typical sequence of steps: initial point cloud data acquisition, object detection, pose estimation, collision-free path and grasp planning, object grasping, and object placement. Most methods in bin-picking assume known geometrical representations (a computer-aided design CAD-model) of the workpiece in question including the specification of admissible grasps for applying template-matching methods [54.40, 41]. Figure 54.11 depicts a variant of a fast template-matching method, which encompasses the following steps for detecting object poses:

- For detecting the scene (e.g., a carrier or box filled randomly with workpieces) typically laser-based sensor systems are used for acquiring a sufficiently dense point cloud. The object pose detection then is considered as a combinatorial optimization problem for which a construction heuristic is applied. For this heuristic tree search, a finite set of possible workpiece poses is initially derived from the search space.
- In order to use a decision tree, the elements of the search set are split into two components: The first component describes a point of interest (POI) in the search space which is part of the workpiece

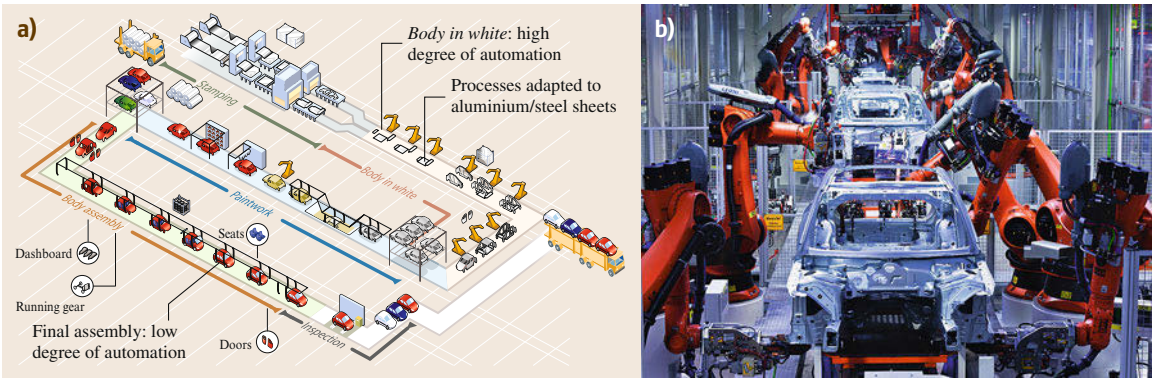


Fig.54.12a,b Car production **(a)** usually follows the illustrated steps along the assembly line: Stamping of the metal sheet into plates, fixing and alignment of the plates on trays, spot welding, painting the car body, and final assembly of the car body (doors, dashboard, windscreens, power-train seats, and tires). Car factories can host over 1000 industrial robots working typically three shifts per day (courtesy of PSA Peugeot Citroën, Paris and Art Movies, Paris). The Audi plant in Ingolstadt Germany **(b)** is highly automated. The picture shows spot welding robots along the *body-in-white* transfer line. Trays carrying car bodies pass through the *robot garden*

surface. The second component describes possible workpiece poses relative to a POI. The partial search quantities obtained thereby have a significant lower complexity compared to the original search set, since the points of interest can provide a constraint on relative workpiece poses, thus restricting their assumed freedom of movement.

- Typical tree search strategies such as best-first search can be used. In that case best-first search explores the search tree by expanding the most promising nodes first. These nodes are chosen according to a heuristic evaluation score, representing the estimated distance from the node to a solution.
- Final evaluation of the workpiece poses is provided by a six-dimensional (6-D) Hough voting procedure, i. e., a generalized Hough transform. The features used for Hough voting are sensor measurements located relative to a POI. For all possible constellations of a sensor measurement relative to a point of interest, a probabilistic statement about possible workpiece poses can be made. Through the superposition of all probability statements, solution candidates can be formed, which are subjected to a statistical test based on a quality rating. The obtained quality rating along with a given level of significance is used in order to decide about the acceptance of a workpiece pose.

The method is able to locate three to four workpieces on average within 0.5–2 s, using a standard desktop computer. Moreover, robot grippers are equipped with a seventh axis to allow grasping parts from corners of the bin. Furthermore, grippers should be formed in such a way that they may reach deep into the bin so that they offer only minimal collision volumes.

54.4.2 Welding

Welding is a manufacturing process that joins materials by applying heat, sometimes with pressure. Usually, workpiece material is melted at the process location often with additional filler material. Typical robot-based welding processes are spot welding, particularly in car body assembly, and gas-shielded metal arc welding (GMAW). With increasing compactness of laser sources and robot motion accuracy, laser welding is in the rise.

Manual welding requires skilled workers, as small imperfections in the weld can lead to severe consequences. Furthermore, welders are exposed to hazardous working conditions (fumes, problematic ergonomic working positions, heat, and noise) so that the use of robots has become beneficial in GMAW processes even for the smallest lot-sizes. Commonly, the automatic arc-welding process is based on a consumable wire electrode and a shielding gas that is fed through a welding gun. Modern welding robots are particularly suited through the following characteristics:

- Computer control allows efficient programming of task sequences, robot motions, external actuators, sensors, and communication with external devices such as welding sources.

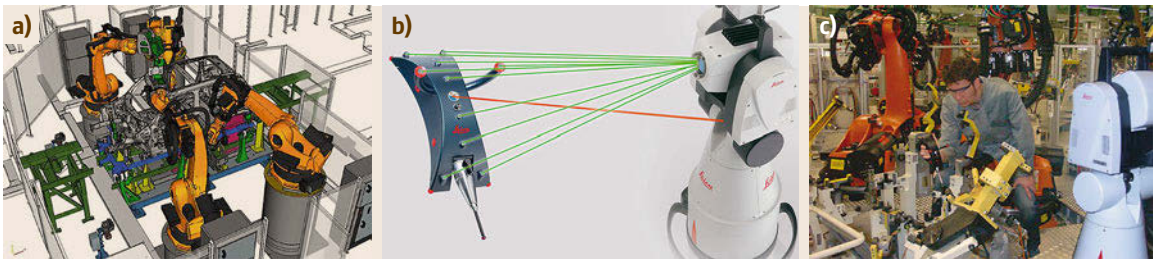


Fig.54.13a–c Offline programming of a spot welding workcell. **(a)** The robot workcell and the task execution are modeled on the basis of realistic robot models (geometry, kinematics, kinetics). **(b)** The shown laser tracker is a portable measurement system that relies on laser beams to accurately measure in a radial volume (accuracy of ± 10 ppm $= 10\,\mu$m/m, up to 80 m in diameter, measuring rate up to 3000 points/s). If measured objects cannot be equipped with reflecting targets or reached by the tracker, handheld probes are tracked instead. **(c)** A tracker in use for interactively measuring the geometry of the robot workcell (courtesy of Leica, now Hexagon MI)

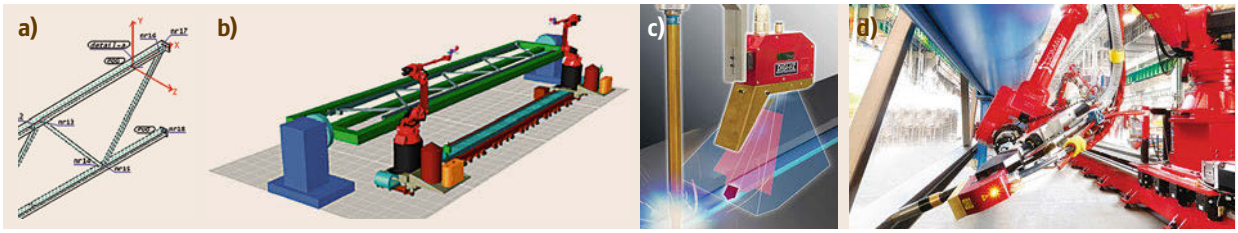


Fig.54.14a–d GMAW welding of building trusses in lot-size one by robot. Illustration **(a)** shows the CAD drawing of a steel truss with relevant information for welding process, **(b)** one-half of the welding workcell with two welding robots working simultaneously on the truss when the neighboring truss on the other half is loaded or unloaded, **(c)** the laser-based seam finding and tracking sensors and **(d)** the welding robot (courtesy of Servo Robot, Canada; Goldbeck, Germany)

- Free definition and parameterization of robot positions or orientations, reference frames, and paths.
- High repeatability and positioning accuracy of paths. Typically repeatability is some ± 0.05 mm and positioning accuracy is better than ± 1.0 mm. These values can be significantly improved through modern robot calibration methods [54.42].
- High speeds of the end-effector of up to 8 m/s for quick approach and depart motions.
- Typically, articulated robots have six DOF so that commanded orientations and positions in their workspace can be reached, which in the welding case means there is one DOF free for rotation around a rotational-symmetric welding tool. Additionally, workspace extensions by mounting the robot on a linear axis (seventh DOF) or even on mobile bases are common, especially for welding of large structures.
- Typical payloads range 6–150 kg. Higher load capacities are required for spot-welding guns (typically > 50 kg) and their cable package.
- Programmable logic controller (PLC) capabilities such that fast input/output control and synchronizing actions within the robot workcell are accomplished.
- Interfacing to high-level factory control through factory communication networks.

Electric current sources, torches, and peripheral devices for automatic cleaning and maintaining the GMAW torch (anti-splatter, wire-cutting, tool changer, etc.) are offered by specialized companies. Often sensors are used to track welding gaps and measure weld seams either before or synchronously with the welding process, thus adapting the robot's trajectory in the presence of workpiece variation and distortion. Also, collaborating robots have been introduced where one robot fixes and moves the workpiece in synchronization with another robot carrying a welding tool so that the weld can be performed with the pool of molten metal horizontal.

Spot Welding in the Automotive Industry with Offline Programming

Car manufacturing has been one of the key drivers in the technical development of industrial robots as the precision handling of spot-welding guns was one of the first breakthrough use cases (Fig. 54.12) *body-in-white* (i. e., unpainted car body) assembly is mostly done by robots, very much in contrast to the final assembly which is

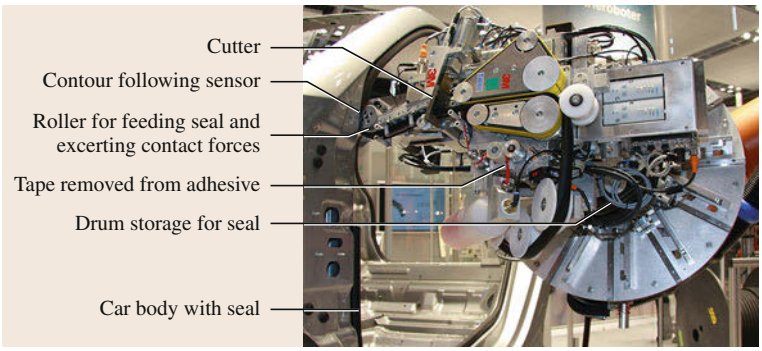


Fig. 54.15 A robot-guided tool for handling and processing limb material, in this case a self-adhesive seal for car bodies

dominated by manual work. Demands for faster cycle times have led to a concurrent and coordinated motion of the spot-welding gun and robot: the robot continues to move while the weld gun is simultaneously rotated about the electrode axis during welding [54.43].

Most of a spot welding robot's programming is done using offline programming (OLP) packages (Fig. 54.13. A library of robots, devices, and advanced CAD capabilities helps plan, program, visualize, and optimize layouts and complete production cycles under assumed manufacturing conditions. Robot programs can be generated and downloaded to robots workcells. A critical step is the calibration of the robot workcell with respect to the simulations [54.44].

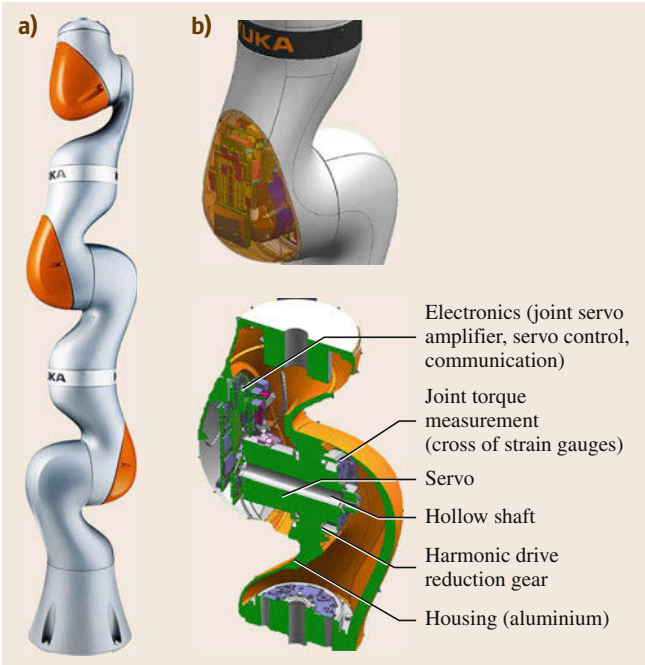


Fig.54.16a,b Design of the KUKA iiwa: **(a)** Shape and **(b)** integration of joint mechatronics

Arc Welding in Metal Construction

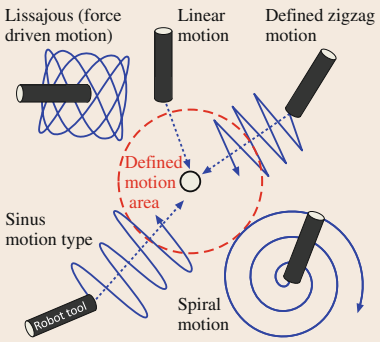
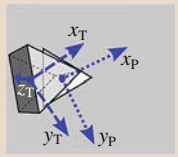
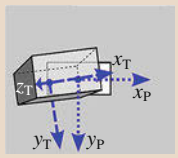
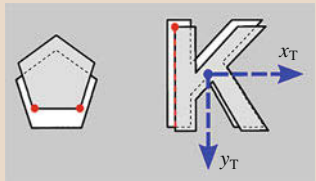
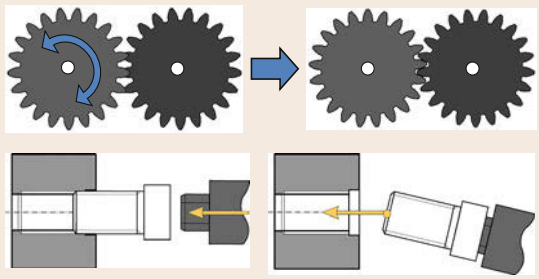
Normally steel constructions are designed using CAD programs that offer functions for GMAW-task definitions such as welding parameters, multipass seams, weld beads sequencing, etc. This information may be used for automatically generating welding robot programs, even in the case of lot-size one jobs.

As an example, the generation principle of a welding robot program is depicted in Fig. 54.14. Large-scale trusses of up to 15 m for large halls are welded-to-measure. The robot program is generated from the CAD drawing with process relevant information. Workpiece tolerances for example, induced by placing the steel components into the fixtures, by bending of the material under its own weight are compensated through active measurement. The robot-mounted sensor locates the weld seam by laser-based vision for shifting the generated programs in such a way that they match the actual weld seams. This calibration is automatically performed if expected and actual bead locations are within a range of ± 2.5 cm.

54.4.3 Assembly

Assembly in manufacturing describes the combination of subsystems or components to systems of a higher complexity through joining. Assembly in manufacturing comprises four process groups: joining, handling, controlling, and auxiliary processes (cleaning, adjustment, marking, etc.) [54.45]. The composition of these four functions may vary depending on batch size, product, and throughput: from assembly workcells to high-throughput assembly lines. Assembly processes form up to 80% of a product's manufacturing cost and this is where the greatest competitive advantage can be gained [54.46]. Therefore, optimization in assembly includes tightly interweaved aspects: Design

Table 54.2 Assembly subprocesses or *modules* and their implementation on the KUKA LBR iiwa

Assembly subprocess	Characterization	Principle
search: search module which supports several search motions or search strategies	Search motion type examples: ● Linear ● Zigzag ● Spiral ● Sinus ● Lissajous Commanding the search motion generation: ● Position-based trajectory ● Force-based trajectory ● Combination of position and force-based trajectory	
peg in hole: execution of typical part joining motions	Reduction of arbitrary joining part types to three abstract planar types: ● Round (arbitrary axial orientation) ● Triangular (defined axial orientation) ● Rectangular (additionally defined workpiece coordination system) Strategy of triangular types is common for numerous workpiece contours: ● Orientation is given, tipping is executed in one step; object pivots around one corner	  Parts T and P of joining motion with local coordinate systems   • *Red dot*: Corner which enters first • *Red dotted line*: edge which follows in tipping
gear, *search*: toothed gear joining motion and *screw-in* motion	● Meshing in toothed gears: Torque oscillations about gear axis and linear forward motion ● Screw-in motion: fixed torque for torque-based screwing ● Angle-controlled tightening required in most screwing process applications: command fixed torque, then turn by a defined angular increment.	

for Assembly (DFA), workcell and assembly line design as well as logistics and manufacturing organization [54.12]. Early on, industrial robots were used in assembly automation, particularly in high-throughput manufacturing lines (Fig. 54.4). However, robots are increasingly used in highly flexible workcells and will enter agile lean manufacturing workplaces as versatile tools at the hands of the human worker. In the following, selective use cases of robots in assembly will be described by detailing on specific enabling technologies.

Assembly of Limb Material

Numerous assembly processes include handling of limb materials such as rubber hoses, wire harnesses, etc. that have to be fixed in position in order to be joined (Fig. 54.15). Obviously stabilizing the material and securing process quality often result in ingenious grip-

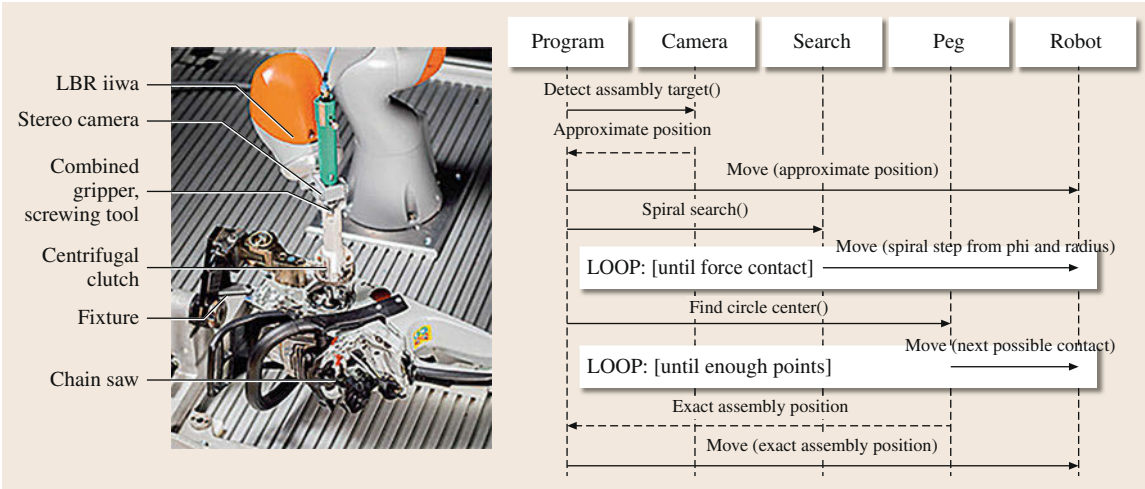


Fig. 54.17 Set up and implementation of a centrifugal clutch assembly for a chain saw with a sequence diagram depicting the consecutive steps until tightening the clutch. Through the robot's torque sensors in each of its links and an appropriate kinematic and dynamic model, the resulting forces at the tool tip are controlled

per designs involving additional actuation and sensing functionalities.

An example is the automatic application of self-adhesive seals as they easily lose their shape and can be stretched or compressed. Since manual application of adhesive seals to vehicle bodies or doors is sensitive and ergonomically problematic a robot-guided tool has to secure bonding of the material's surface to the car body. The seal material is fed from a roll under correct tension and the tape, which covers the adhesive, is removed and stored in a small tank. At the tip of the tool a laser sensor follows the car body or door contour and an actuated roller produces a continuous normal force on the seal. Both the laser sensor and roller's motion are translated into a tension free motion of the robot. In addition, a magazine on a flange ensures that the seal is correctly tensioned and a material reserve for one car door is provided [54.47].

Here, the robot acts as slave to guide a tool which acts as both measuring unit and precision actuator with own master controller. Further efforts aim at embedding rich sensor and control modalities in the robot to account for complex process control based on tactile and geometric information.

Advanced Robot-Based Assembly Process Control

Automation of advanced assembly processes depends on physical contact between the joined workpieces. For this contact formation to be controlled a robot should offer compliant motion control which is a control method that modulates robot position and velocity based on measured or estimated joint torques or contact forces [54.48]. Subject of intense research efforts for a long time application packages for compliant force control in industrial robots which fulfill requirements regarding versatility, robustness, and ease of use in programming have become available during the last ten years [54.49]. The solutions are commonly based on a 6-DOF force–torque sensor which is attached to the robot flange.

The fully torque-controlled DLR (Deutsches Zentrum für Luft- und Raumfahrt) lightweight robot broke new grounds as its 7-DOF redundant kinematic structure, torque sensing in every joint and a variety of compliance modes allowed difficult assembly tasks with complex contact formation during the joining process [54.15]. DLR and KUKA managed to successfully go the strenuous road from the original LBR invention, an idea made manifest in 1991, to a product, first applied in research and predevelopment of new industrial manufacturing concepts in a series of development steps: KUKA LBR3 (2006), LBR4 (2008), and LBR iiwa (2012) [54.25]. Figure 54.16 depicts the integrated mechatronic design of a joint with its unique joint-torque measurement.

To simplify the programming of complex joining processes several assembly subprocess modules were developed of which three are depicted in Table 54.2: the search for contact formation and the execution of two typical joining motions (peg-in-hole, toothed gear joining).

Figure 54.17 depicts a scenario of a force-based centrifugal clutch assembly for a chain saw. The robot

detects and locates the clutch workpieces on a tray by vision sensors. Now the grasped workpiece has to be joined and tightly screwed onto the shaft. In this case the robot's tactile capabilities and compliant behavior are used to achieve a robust and quick assembly. Once a rough shaft center position estimate is acquired by the robot's vision sensor, the robot approaches to this position and starts performing a search pattern until contact detection. The peg-in-hole process is followed by a screw-in motion for tightening the clutch. Alternative designs may be based on 6-DOF industrial robots with a wrist-mounted force–torque sensor to measure process-induced forces on the robot, or simply estimate the forces from the motor torque control [54.50].

The sequence diagram shows a simplified execution trace of a spiral search of a circular shape. Listing 54.1 lists part of the Java code in the KUKA Sunrise control system which identifies the three parts of the compliant assembly: main function, spiral search until collision, and border walk.

Listing 54.1 Extract from the code controlling the sequence of the centrifugal clutch assembly

Main Program:

```
Frame rough = camera.detectAssemblyTarget(); // Retrieve rough position from stereo vision
// Move robot at 30 mm/s. Linear motion. Stop if force > 20 Newton
ForceCondition forceCond = new ForceCondition(robot.getDefaultMotionFrame(), 20);
robot.move(lin(rough).setCartVelocity(30).breakWhen(forceCond)); // First approximation
Frame target = spiralSearch(rough,forceCond); // Obtain the assembly target
assemble(Target); // Assembles the shaft at the given position
```

Search Module. Returns the assembly target:

```
Frame spiralSearch(Frame rough, ForceCondition forceCond){ // Spiral search until collision
 for (double phi = 0; phi < 10.0 * pi; phi +=pi / 90.0){ // 5 loops, 2° step
  Frame spiral = rough.copy();
  Double radius = max_radius * phi / (10.0 * pi);
  spiral.setX(rough.getX + Math.cos(phi) * radius);
  spiral.setY(rough.getY + Math.sin(phi) * radius);
  robot.move(lin(spiral).setCartVelocity(30).breakWhen(forceCond));
  if (motion.getFiredBreakConditionInfo())return findCircleCenter(); // It collides
 }
 return null; // Target not found}
```

Border walk to find shaft center:

```
Frame findCircleCenter()\textbraceleft // Calculates the center of a circular shape
 ArrayList contact = new ArrayList<Frame> (); // Stores frames where collisions take place
 do{
   Frame next = getNextFrame(robot.getForce());
   // Force feedback combined with search pattern such as a grid produces next frame

   robot.move(lin(next).breakWhen(forceCond)); // Move to next position, if possible
   if (motion.getFiredBreakConditionInfo()) contact.add(robot.getDefaultMotionFrame());
  } while (!areEnoughContacts(contact)); // Until enough contact points to identify center
  return getCircleCenter(contact); // Circle center can be determined with three points
}
```

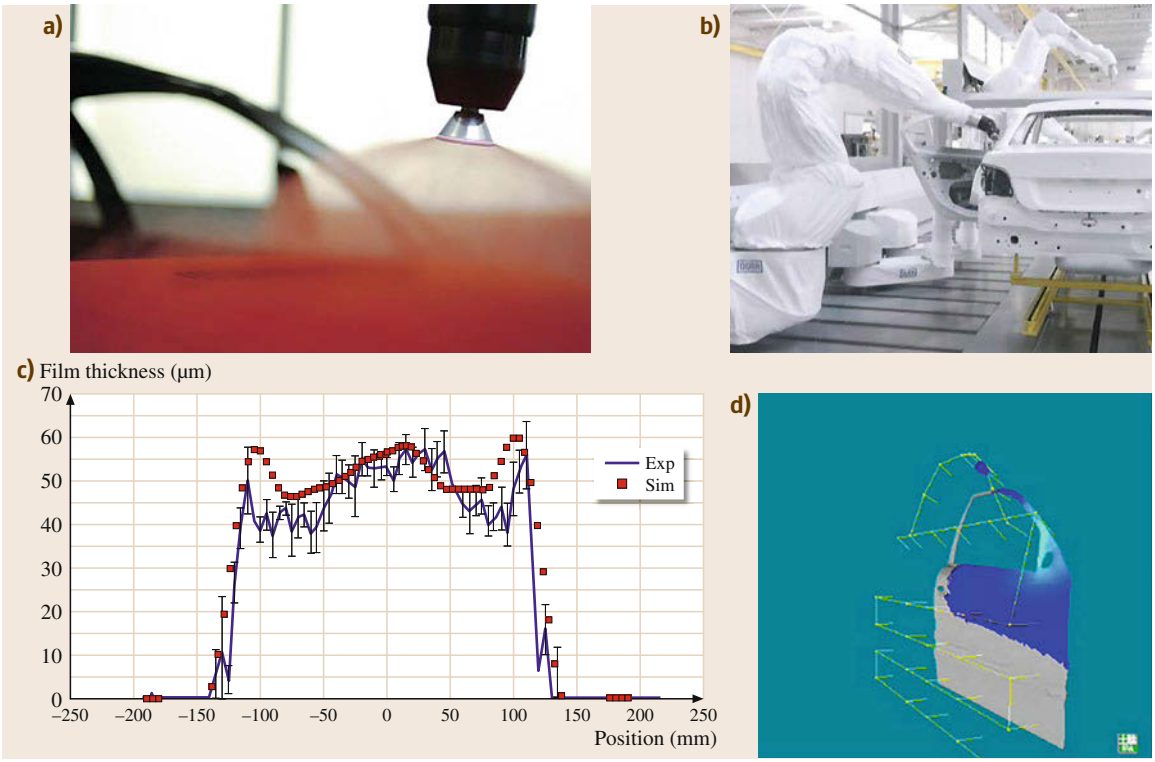


Fig.54.18a–d High-speed rotating atomizer and a multirobot workcell for car body painting. **(a)** A Dürr EcoBell2 paint gun which atomizes the paint material at the edge of the rotating bell disk by centrifugal forces. All current paint materials such as solvent-based or water-borne paints can be used in car production. **(b)** Multiple robots work in parallel for optimal throughput and accessibility of the car body. Most of the programming which includes the synchronization of the robots is performed offline, also with respect to optimal coverage of the painted surface. **(c)** Simulation of the painting process is critical for achieving highest yields (e.g., minimal overspray, uniform paint deposition, etc.) **(d)** Results from simulations for optimized program generation are shown (courtesy of Dürr, Fraunhofer IPA, Stuttgart)

54.4.4 Painting

Hazardous working conditions for human operators motivated Trallfa, a Norwegian company, to develop simple spray-painting robots in 1969, particularly for spraying bumpers and other plastic parts in the automotive industry. Initially pneumatically driven for anti-explosion reasons, today's robot designs are fully electric. They also have hooks and grippers to open hoods and doors during the painting task. Hollow wrists that house gas and paint hoses allow fast and agile motions. Spray guns for robots have evolved dramatically for delivering uniform quality using as little paint and solvent as possible and also for switching between different paint colors. Originally spray-painting robots replicated movements copied from human workers. Most of the programming for robot painting today is done offline as state-of-the-art programming systems offer integrated process simulations to optimize paint deposition, thickness, and coverage (Fig. 54.18). The simulation of the process is quite complex as different effects are taken into account such as turbulent flow field between atomizer and target, static electrical field between rotating bell disk and target, charging of the paint droplets at the bell, space charge effects due to the flow of charged paint droplets, and Coulomb forces acting on the droplets [54.52].

54.4.5 Processing

Material removal processes such as grinding, deburring, milling, and drilling are increasingly carried out by industrial robots with serial kinematics as they combine dexterity, versatility, and cost-effectiveness. The employed process tools are often combined with passive compliance or active force control as the workpiece geometry commonly exhibits tolerances in geometrical or material properties [54.53]. However, robot accuracy (±0.5 mm range) compares poorly to values in the ±0.01 mm range of typical machine

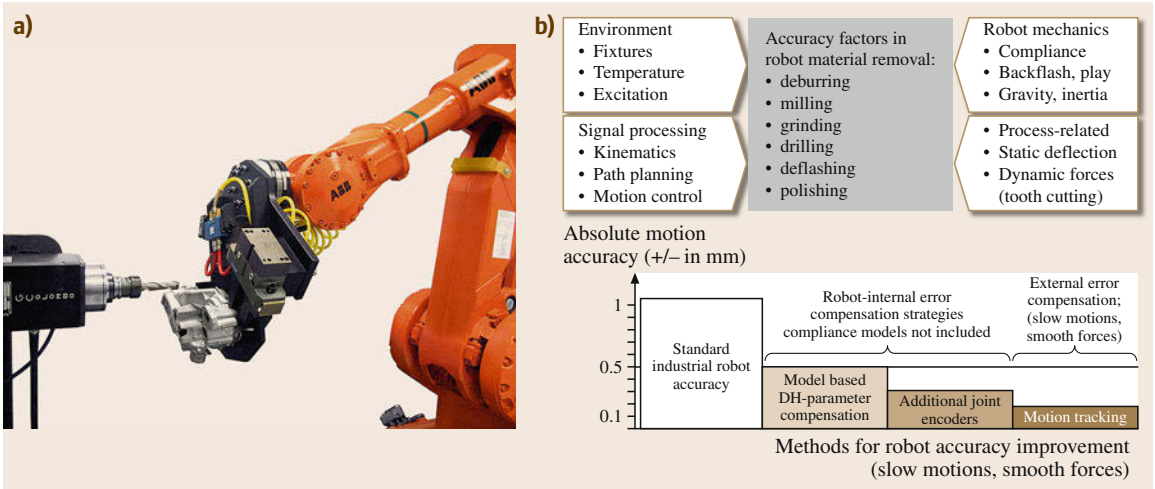


Fig.54.19a–d For small-to-medium-sized parts, a preferred configuration is to have the part grasped and guided by the robot relative to the fixed spindle. **(a)** The robot's gripper is mounted on a force–torque sensor to measure and limit process forces. For machining the edges of the workpiece (deflashing) the machining software package provides a machine-learning technique for optimising the motions [54.51]. **(b)** Influences on accuracy in robotic machining and methods for robot accuracy improvement. Still typical machine tool motion accuracies are in the order of ± 0.01 mm or better (courtesy of ABB, Fraunhofer IPA)

tools [54.54]. The lower eigenfrequencies and damping coefficients of mechanical structures should be as high as possible for precision: The lower eigenfrequencies of milling machines are in the range of 50–100 Hz as compared to 20–30 Hz for typical industrial robots [54.55, 56]. Figure 54.19 depicts the factors that affect the robot's accuracy in typical material removal applications. Figure 54.20 shows first eigenfrequencies, damping, and stiffness in Cartesian space. These characteristics are essential for the design of machining processes and resulting workpiece quality. More robot-guided processes such as laser welding and laser cutting depend on achieving similar accuracies.

Robot position accuracy results from geometric error sources (deviations between actual robot structure and assumed Denavit–Hartenberg (DH) parameters) and nongeometric parameters (compliance of the me-

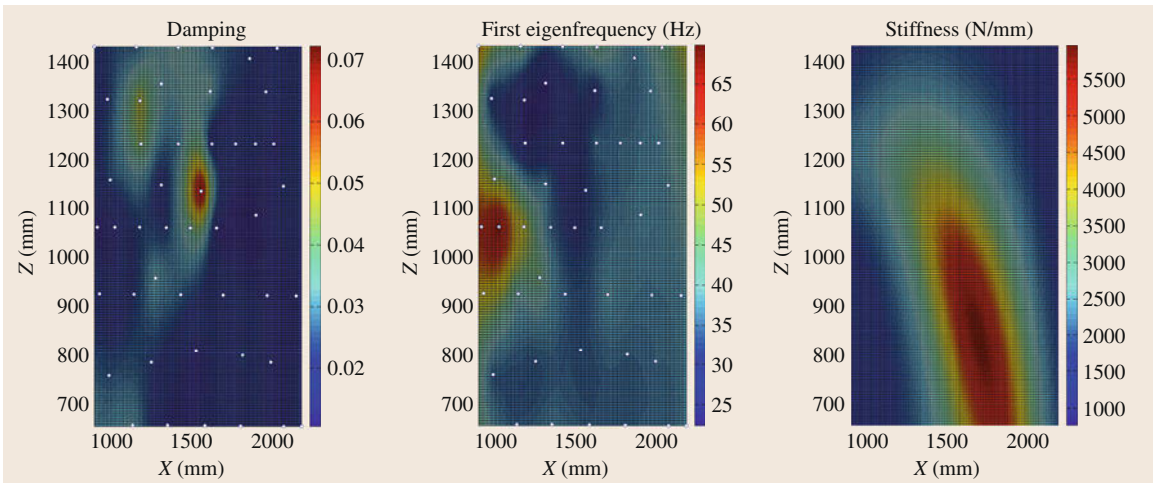


Fig.54.20a–c Robot dynamics in Cartesian space: **(a)** Damping and **(b)** first eigenfrequencies for a KUKA KR60 and **(c)** stiffness for a KR125 in a typical XZ-process plane (measured from the robot's first axis) of the robot's workspace (courtesy of ISG Stuttgart, Fraunhofer IPA)

Table 54.3 Characteristics of computerized numerical control (CNC) and robot controller (RC)

Category	CNC	RC	Interpretation
Targeted application	Machining, material removal	Handling, assembly	A CNC machine tool is single purpose; generally robots are universal machines
Motions	Path-based, complex contour-oriented	Point- or path-based and motion time-oriented	Extended look-ahead of CNC controllers allows detailed path description and adaption on $\gg 100$ via points (< 10 via points in most robot controllers)
Programming	On-site programming based on standardized programming language (G-code), ISO (International Organization for Standardization) 6983, use of computer aided manufacturing (CAM) tools	On-site teaching (teach pendant, editor) based on supplier specific languages, use of typical robot simulation environments	Whereas robots are traditionally programmed manually on site, CNC controllers use CAM technology to generate complex paths automatically based on CAD data
Command reading	Online interpreter, continuous loading of instruction	Initial loading of programs, which are usually interpreted, sometimes compiled)	Program size in robot controllers limited by memory, CNC interprets programs online, may execute an unlimited number of commands

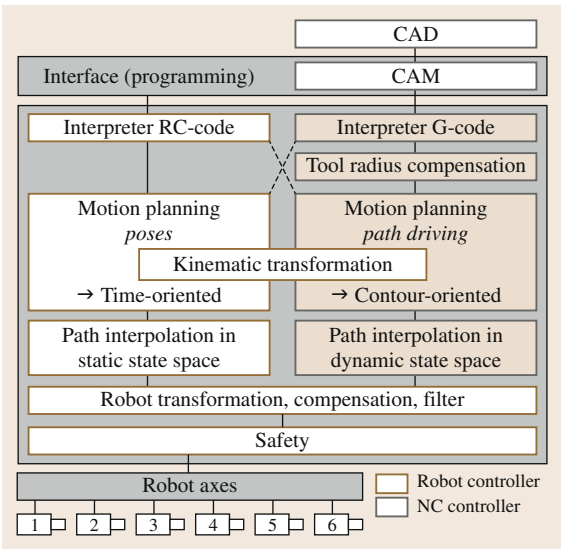


Fig. 54.21 Structure of an NC-kernel integrated into a robot controller (courtesy of ISG Stuttgart)

chanical structure and gear play). In order to reduce the impact of the nongeometric parameters several stepwise approaches can be implemented:

- In combination with force prediction or online measurements robot compliance can be compensated by means of joint stiffness models [54.57]. In drilling applications additional encoders are sometimes mounted on the arm side of robotic joints in order to measure gear-induced joint compliance and backlash, or those effects can be estimated based on determined joint properties [54.58], so that a compensation can be achieved.
- In combination with the geometric error calibration accuracies of better than ± 0.2 mm for typical robots have been realized in larger robot workspaces of $3 \times 3 \times 2\,\mathrm{m}^3$ [54.59].
- For higher accuracies further sensor and actuators systems have been introduced. Using optical tracking deviations of ± 0.2 mm have been demonstrated in steel [54.60]. With additional actuation deviations could be reduced to ± 0.1 mm [54.61].

Robot and CNC machine tool controls may have similar origins, but have taken different paths in developments over the years as depicted in the following Table 54.3.

Increasingly CNC controllers are used for robots in material removal applications for taking advantage of the well-established off-line programming tools in the CNC world and for improving motion accuracy of the robot for complex 3-D contours (Figs. 54.21 and 54.22. Modern robot controllers integrate so-called numerical control (NC) kernels which share components of the robot controller such as user interfaces, kinematic transformations, and safety functions.

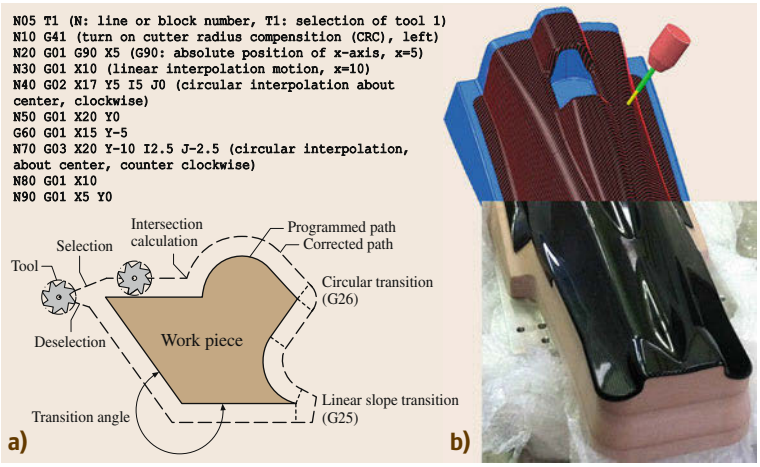


Fig. 54.22 (a) A simple CNC program (G-code, ISO 6983). A CNC program normally is machine independent and orients itself on the workpiece contour after processing. (b) Most CNC programs are generated automatically through CAM tools that transfer geometric information into executable G-code. The example shows generated tool trajectory generation for precision milling of a carbon fiber (CF) part and the finished workpiece (courtesy of Delcam, Birmingham; ISG, Stuttgart)

54.5 Safe Human–Robot Collaboration

Human–robot collaboration allows the combination of typical strengths of robots with some of the numerous strengths of humans. Typical strengths of industrial robots are high stamina, high payload capacity, precision, and repeatability. Strengths of human workers that are unmatched by any machinery comprise flexibility for new production tasks, creative problem-solving skills, and the ability to react to unforeseen situations.

However, industrial robots have a significant potential to harm humans. Therefore, standards for designing and operating industrial robot automation systems have been introduced and found international acceptance. Since 1999, efforts have been made to define measures, rules, and examples specifically for robots in collaborative modes.

54.5.1 Overview of Basic Robot Safety Standards

Machine safety standards provide guidelines for designing and operating any type of machinery. Their consideration is optional, but exhibits the straightforward way of verifying the fulfillment of the fundamental health and safety prerequisites of the Machinery Directive [54.63, 64]. Generally, safety standards are classified into three categories (Fig. 54.23):

- Type A standards: basic standards that define fundamental principles to achieve machine safety.
- Type B standards: generic standards that regulate either a specific safety aspect, e.g., separating dis-

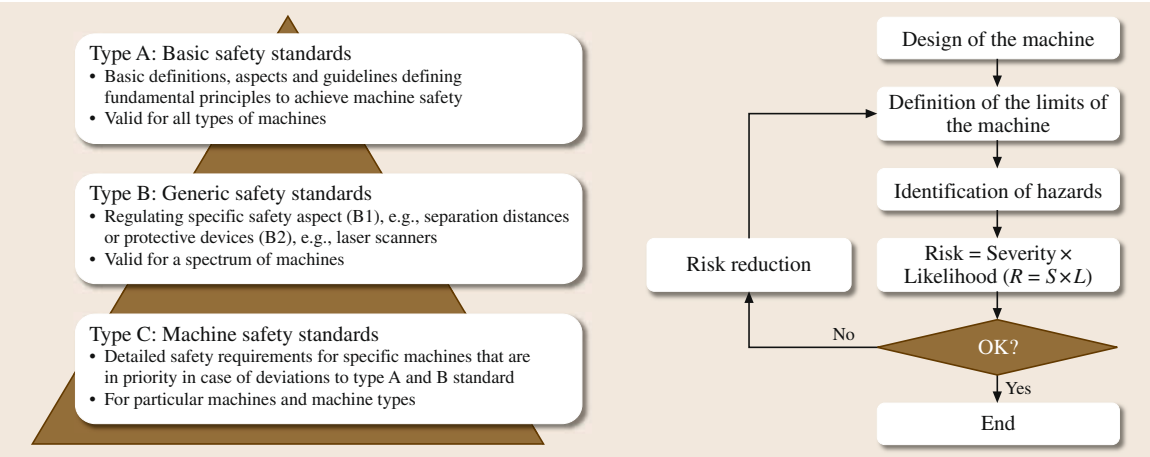


Fig. 54.23 Different types of standards and a simplified iterative risk assessment procedure. See ISO 12100 [54.62, Fig. 1 and 2] for a more detailed schematic representation of risk reduction process

tances, or specific protective devices, e.g., optoelectronic protective equipment like laser scanners or emergency stop (ISO 13850), and are valid for a spectrum of machines.
- Type C standards: machine standards that list detailed safety requirements for a specific type of machine, e.g., industrial robots.

For robotic safety the following standards are of highest importance:

- ISO 12100 [54.62]: Type A standard listing general principles of machine safety, in particular the risk assessment process.
- ISO 13849 [54.65] and IEC 62061 [54.66, 67]: Type B safety standards governing the design of control systems with safety functions.
- ISO 13855 and ISO 13857 [54.68, 69]: Safety distances for separating and nonseparating safety equipment, e.g., fences, light curtains, and laser scanners.
- ISO 10218-1/-2 [54.5, 6]: Type C standards specifically covering safety of industrial robots and robot system integration.

For the setup of any robot installation, an iterative risk assessment process as defined by ISO 12100 has to be conducted. Its workflow as depicted in Fig. 54.23 starts with a functional and geometric design of the machine followed by the definition of the machine limits including spatial boundaries and boundaries regarding usage, e.g., typical tasks, operator qualification, and environmental conditions. The process continues with an identification of the tasks which are consecutively assessed on its risks. Methods for risk estimation are, for example, risk trees as presented in Fig. 54.24. Thereby, any individual hazard is rated on its risk level by quantitatively considering the potential injury severity, the exposition to the hazard, and the possibility to avoid it. The risk estimation leads to a decision if all hazards have been adequately addressed in the machine design. If this is not the case, the machine design is modified to reduce the specific risk and the risk assessment process is repeated.

According to ISO 12100, measures to reduce risks need to be carried out according to the following priorities:

- Risk reduction through inherently safe design
- Risk reduction through safeguards and protective devices
- Risk reduction through information for use (e.g., work instructions, instructions to wear protective equipment).

The central safety standard family for robot safety is ISO 10218. Part 1 addresses the safety requirements of the robots, complemented by part 2, which focuses on the robot system, i. e., the complete integrated machine performing a production task. The second revision of part 1 and the first revision of part 2 were released in 2011 and replace any former robot safety standard, such as EN 775. These standards for the first time define *human–robot-collaboration* as a specific form of a robotic application in an industrial setting and provide guidelines for setting up such collaborative robot systems.

54.5.2 Types and Requirements of Human–Robot Collaboration

ISO 10218-1:2011 contains specific requirements on human–robot-collaboration. It defines four types of collaborative operations in which a human can collaborate with a robot in automatic mode, as depicted in Fig. 54.25. All controller functions for safe human–

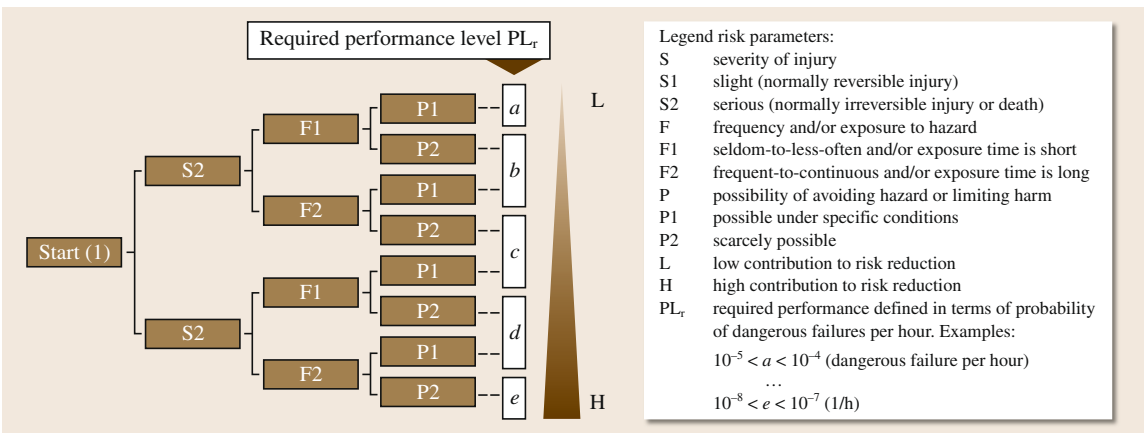


Fig. 54.24 Risk graph for determining required PL_r for safety function according to EN ISO 13849-1 (after [54.65])

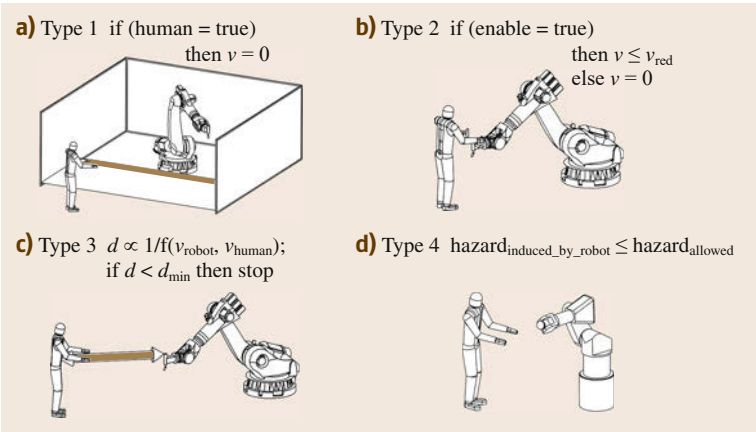


Fig.54.25a–d Modes of human–robot collaboration according to ISO 10218. **(a)** Stop on access with automatic task resumption, **(b)** hand-guiding **(c)** separation and speed reduction, **(d)** monitoring and power and force limiting

robot collaboration have to comply with performance level PL_d with category 3 structures (ISO 13849-1) or SIL 2 (IEC 62061) if the safety assessment does not yield a differing requirement:

- Safety-rated monitored stop (Fig. 54.25 Type 1): The robot is stopped upon access of the human to the collaborative workspace with the robot drives still in control. A so-called safety controller, now offered by most robot manufacturers, assures the standstill of the robot. The robot task can be automatically resumed as soon as the human operator has left the collaborative workspace. Human and robot share the same workspace, but the robot does not move while the human is present.
- Hand-guiding (Fig. 54.25 Type 2): This type of operation implies a direct physical interaction between human and robot with full control of the human over the robot movement. The human guides the robot through a direct input device (e.g., a handle) at or near the end-effector while activating a three-position enabling device (three-position hold-to run device). Thereby the position of the worker within the collaborative workspace is defined. A safety controller for delimiting the robot speed to a specified threshold is required. Hand-guiding in combination with graphic support through icons or 3-D simulation is in particular suitable for intuitive programming of industrial robots [54.70] during automatic mode of the robot.
- Speed and separation monitoring (Type 3): The relative speed between robot and human as well as their distance are actively monitored. If the human is present, the robot has to maintain a safe combination of speed and distance to the human to be able to stop any hazardous motion before a contact with the human may occur. Again safety controllers in conjunction with safe surveillance sensors (including safe sensor data processing) are needed to supervise speed and position of the robot for human

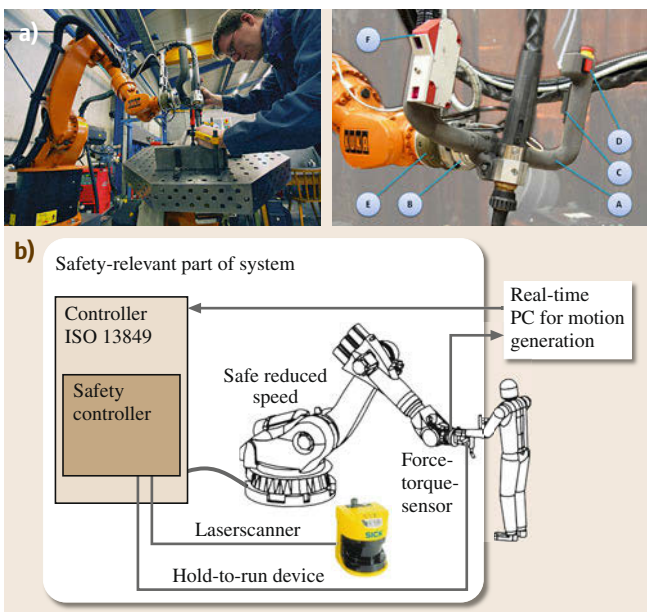


Fig.54.26a,b Intuitive instruction of a welding robot by lead-through programming: **(a)** the worker guides the robot by a flange-mounted (B) handle (A) while pressing safety switches (three-position hold-to run button) (C) on either side. The force exerted on the handle is measured by a force–torque sensor (E) and translated into robot motion. The seam tracking sensor (F) simultaneously measures the workpiece contour for precisely localizing and extracting weld seams. The recorded robot motion is visualized and edited in a simulation environment before task execution. **(b)** The safety relevant part of the system contains a three-position hold-to-run device, a laser scanner for safety monitoring during automatic program execution and a speed monitoring function during hand-guiding is an operational feature which does not have to be realized with a specific safety integrity level (courtesy of Fraunhofer IPA)

motion capturing. Currently, relatively few sensors in industrial automation offer this capability with the required safety integrity level. However, such systems will be available in the future, then being able to feature novel safety strategies for dynamic distance control [54.71].

- Power and force limiting (Type 4): The mechanical hazard potential of the robot is sufficiently reduced to allow a direct, physical interaction of human and robot without an additional safety controller. This is achieved by appropriately limiting collision forces through the design of the robot system such that in the event of a contact between the human and a robot biomechanical tolerance limits are not exceeded.

The ISO 10218 standards in their latest revision as of 2011 explicitly demand that speeds, distances, powers and forces are to be sufficiently limited, but does not give precise threshold values for these limitations. These threshold values need to be determined through risk assessment for a specific application that is foreseen with the robot system. Currently, the technical committee ISO/TC184/SC2 *Robots and robotic devices* develops a technical specification ISO/TS 15066 (in *committee draft* (CD) status of December 2015). It has been drafted to offer more guidance on the risk assessment for collaborative robots and will be released in the near future [54.72]. The procedure to achieve safety outlined in TS 15066 is new for machine safety (not only for robot safety) and is expected to make its way into further standards, especially into the second revision of ISO 10218-2.

For the first time, ISO/TS 15066 introduces tolerance values for the physical strain of the human body. These strain thresholds include maximum forces and maximum pressures for different body parts that can be sustained without suffering from pain or even injury. The risk assessment process involves determining body regions with risk of contact depending on the specific applications and workflows. Based on this information, it can be proved experimentally or in simulation that the given severity threshold values are not exceeded due to limited mechanical or robot control parameters [54.73, 74]. The technical specification aims at transforming highly complex biomechanical injury thresholds into controllable robot performance limits. One of the principles used for dynamic collision analysis is based on the fundamental robot dynamics theory which enables the representation of the reflected inertia of a robot system at any point along the robot structure. Drop-test results from medical injury assessment have been made available for designing robot controllers [54.75].

Part 2 of the ISO 10218 standard provides guidelines regarding the aspects that need to be taken into account during robot workcell setup and system integration. It states the need not only to assess the robot itself, but to take into account the complete application, particularly the end-effector, process, environment, and typical work tasks. This is of special importance for collaborative human–robot operation. As the application always needs to be taken into account, it is not possible to design a *safe robot*, but only to provide robots that are equipped with safety features for setting up a safe collaborative operation.

54.5.3 Examples of Human–Robot Collaboration

In the following, two examples for human–robot collaboration are given.

Intuitive Instruction of a Welding Robot by Lead-Through Programming

Figure 54.26 presents a robot welding workcell which is designed for small batch size production. An intuitive teaching process implementing *hand-guiding* according to ISO 10218 (Type 2 in Fig. 54.25) significantly speeds up robot programming time. The presence of the human during the collaborative operation is monitored via a laser-scanner activating safety zones and safe speed reduction of the robot safety controller during the hand-guiding operation. The recorded motion can be executed automatically after switchover as soon as the human is out of the collaborative workspace.

Collaborative Assembly of Battery-Cases

In this case, sensitive tasks are carried out by the human, while strenuous tasks are executed automatically by a small payload robot (Fig. 54.27 for the cell design). Thereby, the safety concept comprises the requirements for *speed and separation monitoring* (Type 1 in Fig. 54.25) from ISO 10218 and involves two light curtains with a signal processing on a safety PLC and a robot with a safety controller. Switchover between the safety zones is activated upon detection of human presence in a collaborative workplace area. Each safety zone statically monitors the minimum possible distance to the robot's active workspace thereof deriving the maximum possible robot speed. Such a system reduces space requirements for a robot installation while enhancing the flexibility of the assembly process due to the collaborative nature of this assembly process.

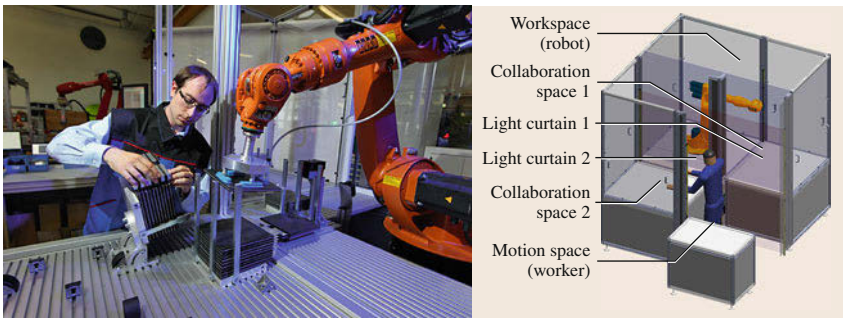


Fig. 54.27 The robot's workcell is divided into three segments by two light curtains which trigger in which area the robot works at lower velocity when collaborating with the worker in the collaboration spaces

54.6 Task Descriptions – Teaching and Programming

Although desirable, a robot cannot be instructed in the same way that one would instruct a skilled human worker how to carry out a task. With skilled human workers knowing the applications, devices, processes, and the general requirements on the product to be manufactured, we would only need to summarize what needs to be done. Humans have, with or without awareness thereof, extensive knowledge about motions, physical effects, cause–effect relationships, learned procedures, etc., and they maintain such knowledge for their own good. A human is skilled because physical capabilities are combined with reuse of that learned/maintained knowledge, and the purpose of the manufacturing operation can be explained and understood.

Robots are not capable of performing such knowledge-based behaviors in a productive manner. Instead, instructions have to be quite explicit and motion oriented, and even motion planning is rarely used since it is difficult to encode much of the required background knowledge. We could aim for programming principles that resemble instructing a (totally) unskilled worker, telling precisely how every aspect of the task is to be performed. Such an explicit way of instructing the robot then should be human friendly, but the performance requirements motivated by productivity needs imply that methods for defining the task need to reflect machine/robot properties considering product data and production processes. Existing approaches include:

- Manually guiding the robot to the positions of interest, or even along the desired paths or trajectories if human accuracy is enough (Fig. 54.26).
- Having simple ways to make use of CAD data whenever available (Fig. 54.13).
- Using different complementary modalities (paths of communication between the human and the robot), such as pointing devices and 3-D graphics.
- Choreographing the task movements, for instance loops and conditions, without requiring extensive programming competencies.
- Means of describing acceptable variation, e.g., acceptable deviations from the nominal path or POI.
- Specification of how external sensing should be used for path adaptations or for handling unknown variations.

Most of these items still mean that the robot is rather dumb, and the knowledge of solutions is maintained among human experts. Consequently, robots are time-consuming and hard to instruct for productive industrial tasks, and the created solutions are not reusable since the knowledge is not explicitly represented in a way that is useful for the robot. To embody knowledge in the robot system is the way to make task definition more efficient, which can be better understood if we review the origins and the progress so far. Initially, mainly during the 1970s and 1980s, there were some painting robots that could be programmed by manual guidance. This was possible due to the following abilities:

- Applications such as painting permitted the use of a lightweight arm, including the end-effector, possible balanced with respect to gravity if needed.
- The accuracy requirements were (compared to today) modest so it was possible to use back-drivable drive trains and actuators, and the definition of the motions could then be done manually by the operator moving the end-effector along the path. The recorded poses, including the timing/speed information, defined the programmed motion.
- Since no inverse kinematics was needed during programming or real-time operation, it was not a problem from a computing point of view to use arm kinematics without singularities in the workspace.

- The requirements of optimality of painting motions were also modest, compared to recent years when environmental conditions (for nature and workers) called for minimal use of paint.

It is often referred to as an inevitable problem that there are singularities to be handled within the workspace. However, to simplify the kinematics and its inverse from a software point of view (e.g., during the 1980s considering the power of the microprocessors and algorithms available at that time), robots were actually designed to have simple (to compute) inverse kinematics. For instance, the wrist orientation was decoupled from the translation by the arm by using wrist axes that intersected with the arm axes. The resulting singularities within the workspace could be managed by restrictions on wrist orientations, but an unfortunate implication was that robot arms were no longer back-drivable (close to the singularities) when designs were (due to engineering and repeatability requirements) adopted to standard industrial controllers. Then with the development of microprocessor-based industrial controllers and the definition of motions based on jogging (manual moves by for instance using a joystick) and CAD data, the means of robot programming became closer to computer programming (extended with motion primitives).

Robot programming languages and environments have traditionally been separated into online programming (using the actual robot in situ) and OLP (using software tools without occupying the robot). With the increasing power of OLP tools, their emerging ability to connect to the physical robot, and the increasing level of software functions that are embedded into the robot control system, online programming is now unusual, except to verify and manually adjust programs generated offline. Of course, it is economically important to minimize downtime for robot programming, and advanced sensor-based applications may be too hard to develop without access to the true dynamics of the physical workcell for fine-tuning. Nevertheless, robot languages and software tools must provide for both methods of programming.

Even though robot languages from different manufacturers look similar, there are semantic differences that have to do with both the meaning when programs are running (the robot performing its operations) and the way the robot is instructed. The need to ensure that existing robot programs can operate with replacement robots and controllers, and also to make use of existing knowledge in robot programmers and incorporated into OLP software, requires manufacturers to continue supporting their original proprietary languages. Features such as backward execution (at least of motion statements) and interactive editing during interpretation by the robot (in combination with restrictions to make the programming simpler) also make robot languages different from conventional computer programming languages.

During the last decade and in current developments, the trend has been to focus on the tool (robot end-effector) and on the process knowledge needed for the manufacturing process, and to let the operator express the robot task in such terms. This development results in a need for an increased level of abstraction to simplify the programming, reflecting the fact that the so-called robot programmer knows the production process very well, but has quite limited programming skills. To understand why such a high level of abstraction has not come into widespread usage, we may compare this with the early days of industrial robotics when there was no kinematics software built into the controllers, and hence the robots were programmed via joint-space motions. (Kinematics here deals with the relation between robot motors/joints and the end-effector motions.)

Built-in inverse kinematics permits tool motion to be specified in Cartesian coordinates, which is clearly a great simplification in many applications. That is, the robot user could focus more on the work to be carried out by the robot and less on the robot itself. However, robot properties such as joint limits cannot be neglected. Until the beginning of the 1990s, robots did not follow programmed trajectories very well at high speeds or accelerations, and full accuracy could only be achieved at low speed. Modern robot systems with high-performance model-based motion control perform their tasks with much greater accuracy at high speed due to model-based control features (Chap. 6).

There are increasing opportunities to raise the level of abstraction to simplify the use of robots even more by encoding more knowledge about the robot, tools, the process, and workcells into control systems. This is a gradual development with possibilities depending on the application; when knowledge is not encoded the robot programmer needs to be more aware of robot properties and how to handle them. Reasons for not including such knowledge typically lie in the fact that it is simply too difficult to do so or because the robot programmer does neither have the competence nor the tools nor reasons to do so. With reference to Table 54.4, the following example explains that type of trade-off and what it means in terms of centricity.

A machine part consisting of steel plates and pipes is to be manufactured by means of welding, and there is robot and process equipment available for production of this (and other) types of workpieces:

- A product-centric system would generate configurations and robot programs, and instruct the oper-

Table 54.4 Robot-related centricity, ranging from a high-level view of the work to be accomplished in terms of the product to manufacture, to a low-level robot motion view that, in practice, constraints what manufacturing operations can be performed

User application ↑ / Robot constraint ↓		Product	Description of the final shape and assemblies of the workpieces, in terms of that product; the robot system plans the operations
		Process	Based on known sequences of specific manufacturing operations, each of these is specified in terms of their processes parameters
		Tool	The motion of the robot-held tool is specified in terms of programs or manual guidance; the user knows the process it accomplishes
		Arm	The robot arm and its end-plate for tool mounting is programmed how to move in Cartesian space
		Joint	For each specified location, the joint angles are specified, so straight-line motions are difficult; the robot provides coordinated servo control



ator for whatever manual assistance that might be needed (e.g. clamping and fixturing). The system would determine the welding data such as the type of welding and how many passes for each seam.

- A process-centric system would instead accept input in terms of the welding parameters to use, including the order of the welds. The system would select input signals to the process equipment (such as what output voltage the robot controller should set such that the welding will be done with a certain current), and robot programs specifying the motions of the welding torch would be generated.
- A tool-centric way of programming would require the operator to set up the process manually by configuring the equipment in terms of their native settings, and appropriate tool data would be configured such that the robot controller can accomplish the programmed motions, which are specified by giving coordinates and motions data referring to the end-effector.
- An arm-centric system is similar to the tool-centric approach in that Cartesian and straight-line motions can be (and need to be) explicitly specified/programmed by the programmer, but extra work is needed since the robot does not support a general tool frame.
- A joint-centric style would be needed if one of the very early robot systems is being used, requiring a straight line to be programmed by lots of joint-space poses close to each other.

Thus, the question is, are you programming robot joint servos to make the robot provide a service for you, are you programming/commanding an arm how to move a tool, is it the tool that is made programmable by means of the robot, is it manufacturing services that are ordered by specifying the desired process parameters, or is it an intelligent system that simply can produce your product? Still today in industry, there are reasons to think and program considering all five levels, although the tool-centric alternative is kind of a baseline. Hence tool-data needs to be maintained in the robot program, as in the next example below.

As a final goal related to the product-centric view, so-called task-level programming is desirable. This has been a goal since the 1980s, and implicitly also since the very beginning of robotics. It would mean that the user simply tells the robot what should be done in common terms and the robot would know how to do it, but it would require extensive knowledge about the environment and so-called *machine intelligence*. The need for extensive modeling of the environment of the robot is well known. Sensing of the environment is costly, but in an industrial environment it should only be needed occasionally. Modeling has to encompass full component dynamics and the limitations of the manufacturing process. With these difficulties, task-level programming is not yet achievable in practice. If we limit our domain, however, such as to machining where motion planning is already used for so-called machine tools, generating the robot programs from product descriptions is feasible since the process (with tolerances due to robot compliance) is modeled.

As indicated in the application examples, it is now common practice to generate robot programs from geometric data in CAD files. That is, the CAD application could be the environment used for specifying how the robot should perform the required operations on the specified parts, which is the CAM part of CAD/CAM (Computer Aided Manufacturing, often in term of a back-end to the CAD system, and hence usually not explicitly referred to). CAM is not quite task-level programming since human operators do the overall planning. CAD software packages are powerful 3-D tools that are very common among manufacturing companies. Consequently, using these tools for robot programming is desirable since the operator may start the OLP of the necessary manufacturing operations us-

ing the 3-D model of the product, even before selecting a robot.

There is an ongoing competition along the so-called vertical integration from product data down to machine operations, including at least the following three approaches:

- CAD providers increasingly offer advanced robot-aware functions, thereby expanding their field of service and expertise from a higher (CAD) to a lower (robot) level. An example of a robot-aware CAM system is depicted in Fig. 54.22.
- Robot providers have developed programming tools with CAD data import and various motions planning modules, and thus, expanded their offerings to higher levels. One such software is shown in Fig. 54.29.
- A further development is to include a CNC kernel into the robot controller, as suggested in Fig. 54.21, making the robot behave as a CNC machine.

These approaches as such are more complementary than competing, with the competing stakeholders presently exploring what benefits can be gained for customers and business in each type of application.

Listing 54.2 Example pick-and-place operation in the ABB robot programming language Rapid

```
PROC PickInPallet()
 MoveJ Offs(pPickInPallet,-500,0,500),v2000,z100,tGripper\WObj:=wobjPalletStatic;
 MoveL RelTool(pPickInPallet,0,0,-100),v2000,z100,tGripper\WObj:=wobjPallet;
 MoveL RelTool(pPickInPallet,0,0,0),v100,fine,tGripper\WObj:=wobjPallet;
 Grip;
 MoveL RelTool(pPickInPallet,0,0,-100),v500,z10,tGripper\WObj:=wobjPallet;
 MoveJDO Offs(pPickInPallet,-500,0,500),v2000,z100,tGripper\WObj:=wobjPalletStatic,
 doNewObject,1;
ENDPROC

PROC PlaceAtOutFeeder()
 MoveJ Offs(pDropOutFeeder,0,0,200),v2000,z10,tGripper\WObj:=wobjOutFeeder;
 WaitDI diOutFeederReady,1;
 MoveL Offs(pDropOutFeeder,0,0,0),v200,fine,tGripper\WObj:=wobjOutFeeder;
 Release;
 MoveLDO Offs(pDropOutFeeder,0,0,200),v500,z10,
 tGripper\WObj:=wobjOutFeeder,doStartOutFeeder,1;
 MoveL pFromOutFeeder,v2000,z10,tGripper\WObj:=wobjMachine;
ENDPROC

PROC Grip()
 SetDO doGrip,1;
 WaitDI diGripped,1;
ENDPROC
PROC Release()
 SetDO doGrip,0;
 WaitDI diGripped,0;
ENDPROC
```

Avidbots Ex1016
Page 1435 of 2154

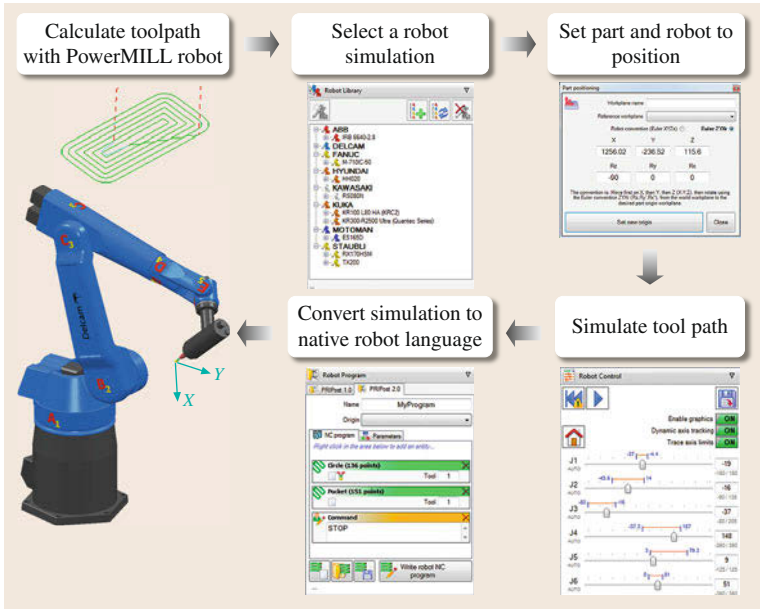


Fig. 54.28 Workflow according to the PowerMill software from Delcam with its robot module. Note that step 4 includes also simulation of robot properties such as closeness to singularities

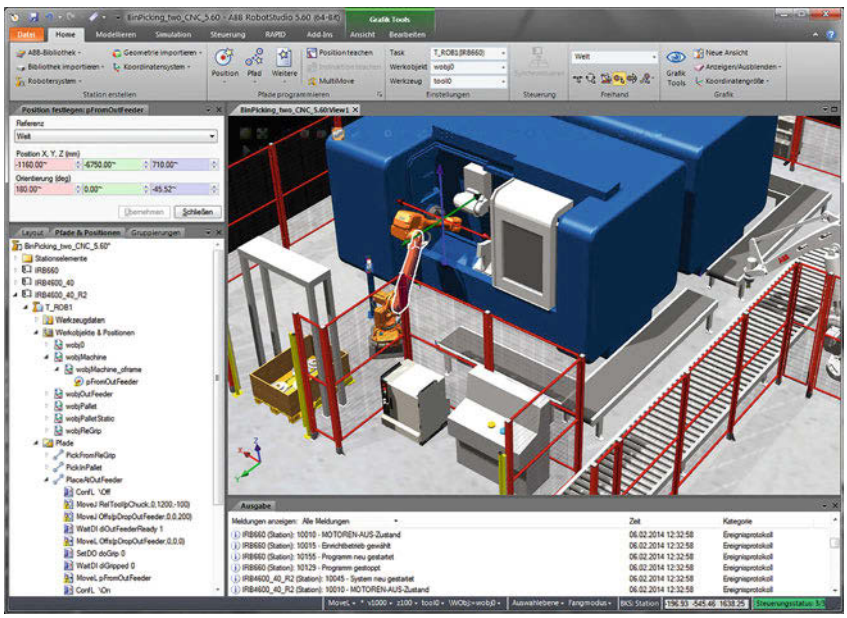


Fig. 54.29 Robot programming environment, with a virtual controller (console to the lower right) for each robot including the model-based embedded control software compiled for the personal computer (PC). To the left there are the instances of all objects depicted as a tree structure, ranging from complete machines down to the bits of the I/O signals

At step 5 in Fig. 54.28, the motion specification is generated and expressed in the native language of the robot. On the other hand, the tool path could be expressed in machine-independent, standardized G-code. A further step would be to integrate a machine tool interface and control into the robot controller which is depicted in Fig. 54.22. What approach to take is an area of current research and development (R&D) and business decisions, but most robots are programmed in their own language only, since the needed system functions as a consequence of Table 54.4.

To exemplify task definitions based on a native robot language, consider the application depicted in Fig. 54.29, which includes bin-picking, handling, machine tending, conveyer tracking, and palletizing. Listing 54.2 lists a sample program with some of the functions from that application programmed in ABB's proprietary robot language Rapid. The program being

tool-centric shows in the tool argument of the Move statements, the *tGripper* that is a reference to the tool data including the tool frame. At the same time, it is arm-centric in the way that the execution of motions is performed per arm (with support for synchronized execution for multiarm robots), as expressed by the *MoveL* statements. Correspondingly, the *MoveJ* performs a joint-space motion.

Note that Grip and Release are in separate procedures where execution waits until a hardware acknowledged signal is obtained. For Move statements L denote Linear, J means Jointspace, functions Offs and RelTool compute target poses based on other determined poses (i. e., frames). The v-constants are velocities in mm s^{-1}, and z-constants (or fine) specify how close to the target pose the motion should be. DI refers to digital input and DO to digital output. The WObj is the work object, i. e., the specification of the base frame of the motions.

When programming motion behaviors like a search strategy, which could be considered being a motion primitive from an application point of view, use of the end-user language of that robot might not be the best option. Instead a computer programming language can be more appropriate as shown in Listing 54.1. In the ABB case, the strategies are implemented as part of an application package as for assembly, and the code is internally written in C (cannot be viewed or changed by the user, but has optimized real-time performance) while the functionality is exposed to the robot user via special statements like FCRefSpiral [54.51] and more to accomplish the principles of Table 54.2. A variety of solutions are available for different robot brands [54.76].

Considering a complete setup with multiple robots, either with a few robots and some peripherals like in Fig. 54.9 or a complete manufacturing line as in Fig. 54.12, there are several robot programs that need to be programmed to work together and hence there is a coordination and complexity problem. A useful approach is to use a service-oriented approach, considering robots as servers providing services according to a set of programs that are exposed on the factory or production-cell network which means programming is done by:

- Building services (using several technologies) which are available, discoverable or not, remotely accessible by the application programmer
- Building applications that coordinate and use those services.

For such system to be programmable, however, the equipment (typically from different suppliers) has to be configured and interconnected according to the sometimes overlooked art of robot system integration.

54.7 System Integration

It is interesting to note that connecting different workcell devices with each other, and integrating them into a working system, is hardly mentioned in the robotics literature. Nevertheless, in actual nontrivial installations, this part typically represents, apart from the cost for peripheral device, roughly half of the overall installation costs. The automation scenario includes all the problems of integrating computers and their peripherals, plus additional issues that have to do with the variety of (electrically and mechanically incompatible) devices and their interaction with the physical environment (including their inaccuracies, tolerances, and unmodeled physical effects such as backlash and friction). The number of variations is enormous so it is often not possible to create reusable solutions. In total, this results in a need for extensive engineering to put a robot to work (Table 54.5). This engineering is what we call system integration.

Carrying out system integration according to current practice is not a scientific problem as such (although how to improve the situation is), but the obstacles it comprises form a barrier to applying advanced sensor-based control for improved flexibility, as

Table 54.5 Stages of system integration, typically carried out in the order listed

Physical	Selecting equipment based on dimensioning for mechanical size, load, and stress
	Mechanical interfacing (locations, adapter plates, etc.)
	Electrical power supply (voltages and currents for robots, effectors, feeders, etc.)
	Connections for analog signals (shielding, scaling, currents, binary levels, etc.)
	Safety design and risk assessment
Communication	Interconnections for single-bit digital I/O
	Byte-wise data communication, including latencies and bit rates
	Transfer of byte sequences
Configuration	Configuration of messages between interacting devices
	Establishment of services
	Tuning for performance and resource utilization
Application	Definition of application-level functions/services
Task	Application programming, using the application-level services

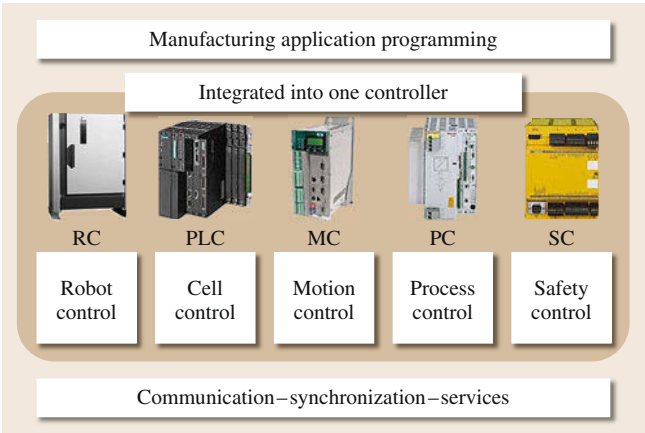


Fig. 54.30 Manufacturing cell with a multicontroller architecture with dedicated controllers (robot motions, production sequences, positioning systems, processes and worker safety). In modern robot controllers, such as the KUKA KR C4, these controllers are replaced by software tasks of a single multifunctional robot controller, thus reducing investment costs on controllers and communication, simplifying engineering, programming and diagnosis, and enhancing process quality and shortening cycle time ►

is needed in small series production. In particular, in future types of applications using external sensing and high-performance feedback control within the workcell, system integration will be an even bigger problem since it includes tuning of the feedback too.

For large series production (e.g. in the automotive industry), the engineering cost of system integration is less of a problem since its cost per manufactured part is small. On the other hand, the trend toward smaller series

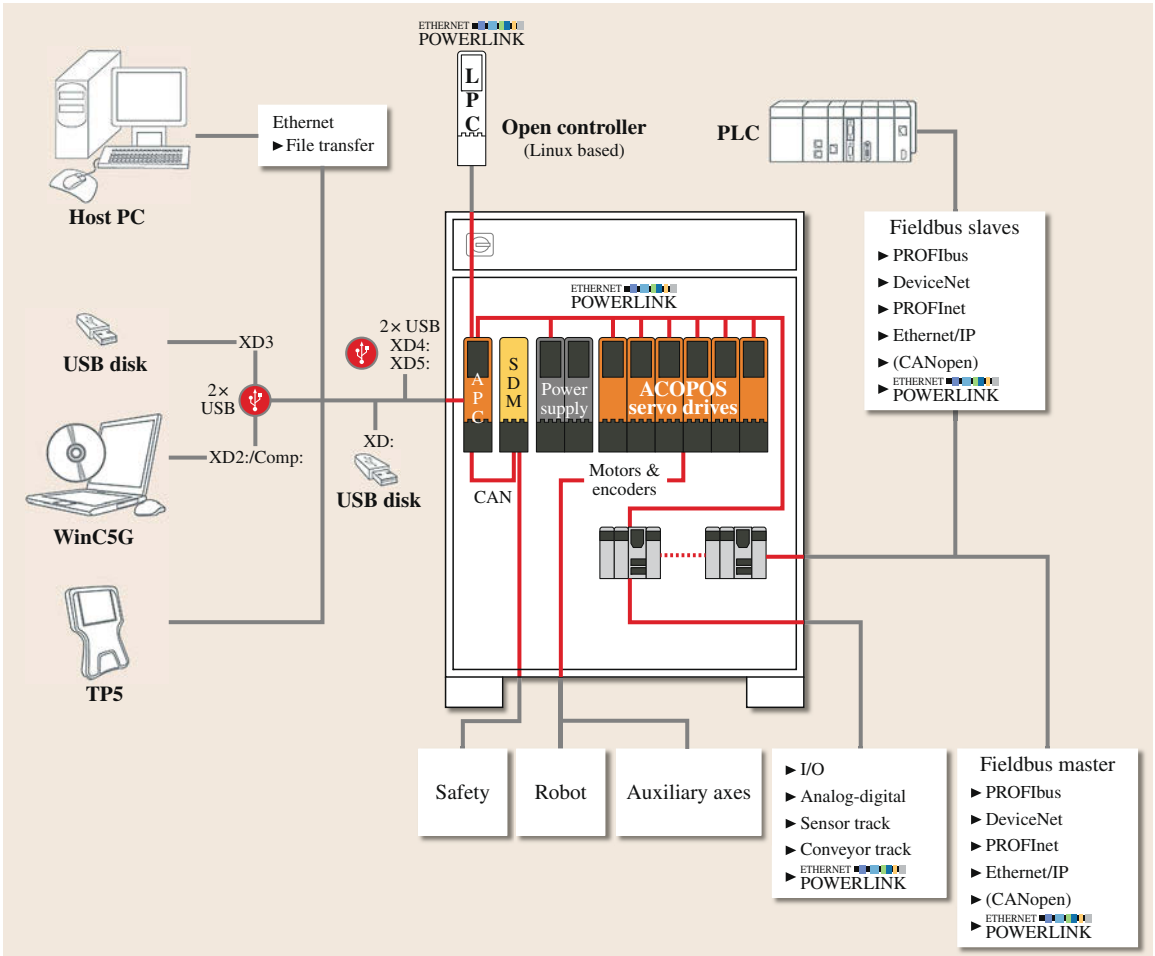


Fig. 54.31 Control modules and interfaces in a typical (but open) robot controller. Several of the named network technologies are based on Ethernet, ranging from normal LAN to different real-time protocols (courtesy of COMAU)

of customized products, or products with many variants that are not kept in stock, calls for high flexibility and short changeover times. Flexibility in this context refers to variable product variants, batch sizes, and process parameters. In particular, this is a problem in small and medium-sized productions, but the trend is similar for larger enterprises as flexibility requirements are continuously on the rise. One may think that simply by using standards for input/output (I/O) and well-defined interfaces, integration should be just a matter of connecting things together and run the system.

Of course, the object-oriented software solution would be to have a gripper class with operations grasp and release. That would be appropriate for a robot simulator in pure software, but for integration of real systems such encapsulation of data (abstract data types) would introduce practical difficulties because:

- Values are explicit and accomplished by external (in this case) hardware, and for testing and debugging we need to access and measure them.
- Online operator interfaces permit direct manipulation of values, including reading of output values.

The use of functions of object methods (so-called set:ers and get:ers) would only complicate the picture; maintaining consistency with the external devices would be no simpler.

Already in small installations, the complexity of system integration becomes apparent. So-called vertical integration means integration of low-level devices with a high-level factory control system, whereas integration of peripherals on (for instance) workcell level is called horizontal integration (Fig. 54.30).

Lack of self-descriptive and self-contained data descriptions that are also useful at the real-time level further increases the integration effort since data interfaces/conversions typically have to be manually written. In some cases, as illustrated in Fig. 54.31, there are powerful software tools available for the integration of the robot user level and the engineering level [54.77]. The fully (on all levels) integrated and nonproprietary system such as the recently discussed Industry4.0 or industrial Internet initiatives is still is a challenge, particularly for small and medium-sized productions [54.24].

54.8 Outlook and Long-Term Challenges

The widespread use of industrial robots in standard, large-scale production such as the automotive industry, where robots perform repetitive tasks in well-known environments, resulted in the common opinion that industrial robotics is a solved problem. This opinion was underpinned by the robot systems' impressive automatic performance, based on advanced semiautomatic programming and resulting in an unbeatable product quality when compared to manual labor. However, large-scale production comprises only a minor part of the work needed on an industrial scale in any wealthy society, especially considering the number of companies and the variety of applications and processes that could and should be automated for productivity, health and sustainability reasons.

Global prosperity and wealth requires resource-efficient and human-assistive robots. The challenges today are to recognize and overcome the barriers that are currently preventing robots from being more widely used, especially in small and medium-sized manufacturing.

Taking a closer look at the scientific and technological barriers, we find the following challenges:

- *Human-friendly task specification*, including intuitive ways of expressing permitted/normal/expected variations and errors. There are numerous upcoming and promising techniques for user-friendly human–robot interaction (such as speech, gestures, manual guidance, and so on), but the focus is still on specification of the nominal task rather than the complete task which is capable of handling foreseen deviations from the nominal case (Part G, Robots and Humans). Taking care of these expected deviations can account for up to 80% of the total programming time.
- *Intuitive human–robot interaction* The foreseen variations, and the unforeseen variations experienced during robot work, are difficult to manage. When instructing a human, he/she has an extensive and typically implicit knowledge about the work and the involved processes. To teach a robot, it is an issue both how to realize what the robot does not know, and how to convey the missing information efficiently. Furthermore, a person should feel comfortable and familiar with the robot's functionality and operation during all operational modes including maintenance and error recovery. Research and development toward industrial robotics usability and ergonomics for increasing acceptance and operator efficiency has been of surprisingly low activity in the past.
- *Efficient mobile manipulation*. Successful implementations and systems are available for both mobility and for manipulation, but accomplished

in different systems and using different types of (typically incompatible) platforms. A first step would be to accomplish mobile manipulation at all, combining all degrees of freedom of both the mobile base and arm(s), intelligently exploiting system redundancies to achieve a given task. A second step providing truly dependable autonomous navigation (with adaptive, but predictable understanding of constraints, such as frequent environmental changes and typical shop-floor dynamics, and appropriate sensor-based reactions) dexterous manipulation with multifingered grippers, and robust force/torque interaction with environments (that have unknown stiffness). As a concluding step, all this needs to be done with decent performance using reasonably priced hardware, and with human–robot interfaces according to the previous items.

- *Low-cost components including low-cost actuation*. Actuation of high-performance robots represent about two-third of the overall robot cost, and improved modularity often results in a higher total hardware cost (due to less opportunities for mechatronic optimization). On the other hand, cost-optimized (with respect to certain applications) systems result in more-specialized components and smaller volumes, with higher costs for small series production of those components. Since future robotics and automation solutions might provide the needed cost efficiency for small series customized components, we can interpret this as a boot-strapping problem, involving both technical and business aspects. The starting point is probably new core components that can fit into many types of systems and applications, calling for more mechatronics research and synergies with other products.
- *Composition of subsystems*. In most successful fields of engineering, the principle of superposition holds, meaning that problems can be divided into subproblems and that the solutions can then be superimposed (added/combined) onto each other such that the total solution comprises a solution to the overall problem. These principles are of key importance in physics and mathematics, and within engineering some examples are solid-state mechanics, thermal dynamics, civil engineering, and electronics. However, there is no such thing for software, and therefore not for mechatronics (which includes software) or robotics (programmable mechatronics) either. Thus, composition of un-encapsulated subsystems is costly in terms of engineering effort. Even worse, the same applies to encapsulated software modules and subsystems. For efficiency, system interconnections should go directly to known (and hopefully standardized) interfaces, to avoid the indirections and extra load (weight, maintenance, etc.) of intermediate adapters (applying to both mechanics for end-effector mounting and to software). Interfaces can be agreed upon, but the development of new versions typically maintains backward compatibility (newer devices can be connected to old controller), while including the reuse of devices calls for mechanisms for forward compatibility (automatic upgrade based on meta information of new interfaces) to cover the case that a device is connected to a robot that is not equipped with all the legacy or vendor-specific code.
- *Embodiment of engineering and research results*. Use or deployment of new technical solutions today still starts from scratch, including analysis, understanding, implementation, testing, and so on. This is the same as for many other technical areas, but the exceptional wide variety of technologies involved with robotics and the need for flexibility and upgrading makes it especially important in this field. Embodiment into components is one approach, but knowledge can be applicable to engineering, deployment, and operation, so the representation and the principle of usage are two important issues. Improved methods are less useful if they are overly domain specific or if engineering is experienced to be significantly more complicated. Software is imperative, as well as platform and context dependent, while know-how is more declarative and symbolic. Thus, there is still a long way to go for efficient robotics engineering and reuse of know-how.
- *Open dependable systems*. Systems need to be open to permit extensions by third parties, since there is no way for system providers to foresee all upcoming needs in a variety of new application areas. On the other hand, systems need to be closed such that the correctness of certain functions can be ensured. Extensive modularization in terms of hardware and supervisory software make systems more expensive and less flexible (contrary to the needs of openness). Highly restrictive frameworks and means of programming will not be accepted for widespread use within short-time-to-market development. Most software modules do not come with formal specification, and there is less understanding of such needs. Thus, systems engineering is a key problem.
- *Sustainable manufacturing*. Manufacturing is about transformation of resources into products, and productivity (low cost and high performance) is a must. One aspect of sustainability deals with energy-efficiency of the devices that are used for production. Saving energy can constrain the path planning of an

individual and a whole fleet of robots, and even shift the operation of energy-consuming tasks in periods of the day where energy cost are low. A second aspect of sustainability deals with recycling of scarce resources in terms of materials and noble earths. In most cases, this can be achieved by crushing the product and sorting the materials, but in some cases disassembly and automatic sorting of specific parts are needed. There is, therefore, a need for robots in recycling and de-manufacturing. Based on future solutions to the above items, this is then a robot application challenge.

An overall issue is how both industry and academia can combine their efforts such that sound business can be combined with scientific research so that future developments overcome the barriers that are formed by the above challenges.

Video-References

VIDEO 260 SMErobotics project video
available from http://handbookofrobotics.org/view-chapter/54/videodetails/260

VIDEO 261 SMErobot video coffee break (English)
available from http://handbookofrobotics.org/view-chapter/54/videodetails/261

VIDEO 262 SMErobot final project video
available from http://handbookofrobotics.org/view-chapter/54/videodetails/262

VIDEO 265 SMErobot – New parallel kinematic with unique concepts for demanding handling and process applications
available from http://handbookofrobotics.org/view-chapter/54/videodetails/265

VIDEO 266 SMErobot D4 *The woodworking assistant*
available from http://handbookofrobotics.org/view-chapter/54/videodetails/266

VIDEO 380 SMErobotics demonstrator D1 assembly with dual-arm industrial manipulators
available from http://handbookofrobotics.org/view-chapter/54/videodetails/380

VIDEO 381 SMErobotics demonstrator D2 human–robot cooperation in wooden house production
available from http://handbookofrobotics.org/view-chapter/54/videodetails/381

VIDEO 382 SMErobotics demonstrator D3 assembly with sensitive compliant robot arms
available from http://handbookofrobotics.org/view-chapter/54/videodetails/382

VIDEO 383 SMErobotics demonstrator D4 welding robot assistant
available from http://handbookofrobotics.org/view-chapter/54/videodetails/383

References

54.1 The International Federation of Robotics (IFR): *World Robotics 2015. Statistics, Market Analysis, Forecasts, Case Studies and Profitability of Robot Investments* (International Federation of Robotics, Frankfurt 2015)

54.2 D. Bourne: My Boss the Robot, Scientific American **308**(5), 37–41 (2013)

54.3 M.P. Groover: *Automation, Production Systems, and Computer-Integrated Manufacturing*, 3rd edn. (Prentice-Hall, Upper Saddle River 2007)

54.4 International Organization for Standardization: Standards catalogue, 25.040.30: Industrial robots. Manipulators, http://www.iso.org/iso/home/store/catalogue_ics/catalogue_ics_browse.htm?ICS1=25&ICS2=40&ICS3=30 (2014)

54.5 International Organization for Standardization: ISO 10218-1:2011. Robots and robotic devices – Safety requirements for industrial robots – Part 1: Robots, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=51330 (2011)

54.6 International Organization for Standardization: ISO 10218-2:2011. Robots and robotic devices – Safety requirements for industrial robots – Part 2: Robot systems and integration, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=41571 (2011)

54.7 L. Westerlund: *The Extended Arm of Man. A History of the Industrial Robot* (Informationsförlaget, Stockholm 2000)

54.8 D. Hunt: *Industrial Robotics Handbook* (Industrial, New York 1983)

54.9 V.D. Scheinman: Design of a Computer Controlled Manipulator, Ph.D. Thesis (Stanford Univ., Stanford 1969)

54.10 International Federation of Robotics (IFR): History of Industrial Robots. From the first installation until today (International Federation of Robotics, Frankfurt 2012), http://www.ifr.org/uploads/media/History_of_Industrial_Robots_online_brochure_by_IFR_2012.pdf

54.11 H. Makino: Assembly robot, US4341502 A (1982)

54.12 G. Boothroyd, P. Dewhurst, W. Knight: *Product Design for Manufacture and Assembly*, 3rd edn. (CRC, Boca Raton 2011)

54.13 R. Curran, G. Mayer: The architecture of the Adept0ne® direct-drive robot, Am. Control Conf., Boston (1985)

54.14 X.-J. Liu, J. Wang: Performance evaluation of parallel mechanisms. In: *Parallel Kinematics*, Springer Tracts in Mechanical Engineering, (Springer, Berlin, Heidelberg 2014) pp. 185–238

54.15 S. Haddadin, S. Parusel, R. Belder, A. Albu-Schaeffer: It is (almost) all about human safety: A novel paradigm for robot design control and planning, Lect. Notes Comput. Sci. **8153**, 202–215 (2013)

54.16 R. Bloss: Innovations like two arms, cheaper prices, easier programming, autonomous and collaborative operation are driving automation deployment in manufacturing and elsewhere, Assem. Autom. **33**(4), 312–316 (2013)

54.17 S. Kock, T. Vitor, M. Björn, H. Jerregard, M. Källmann, I. Lundberg: Robot concept for scalable, flexible assembly automation, Proc. IEEE Int. Symp. Assem. Manuf. (ISAM), Tampere (2011)

54.18 S. Bogh, M. Hvilshoj, M. Kristiansen, O. Madsen: Identifying and evaluating suitable tasks for autonomous industrial mobile manipulators (AIMM), Int. J. Adv. Manuf. Technol. **61**(5-8), 713–726 (2012)

54.19 C.Y.K. Smith, L. Nalpantidis, X. Gratal, P. Qi, D. Dimarogonas, D. Kragic: Dual arm manipulation – a survey, Robot. and Auton. Syst. **60**(10), 1340–1353 (2012)

54.20 P. Gorle, A. Clive: *Positive Impact of Industrial Impacts on Employment* (International Federation of Robotics, Frankfurt 2013)

54.21 J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, A. Marrs: Disruptive technologies: Advances that will transform life, business, and the global economy, http://www.mckinsey.com/insights/business_technology/disruptive_technologies (2013)

54.22 E. Westkämper: *Towards the Re-Industrialization of Europe: A Concept for Manufacturing for 2030* (Springer, Berlin, Heidelberg, 2014)

54.23 M. Hedelind, M. Jackson: How to improve the use of industrial robots in lean manufacturing systems, J. Manuf. Technol. Manag. **22**(7), 891–905 (2011)

54.24 H. Kagermann, W. Wahlster, J. Helbig: *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0*. http://www.acatech.de/fileadmin/user_upload/Baumstruktur_nach_Website/Acatech/root/de/Material_fuer_Sonderseiten/Industrie_4.0/Final_report__Industrie_4.0_accessible.pdf (Forschungsunion/acatech, Frankfurt 2013)

54.25 R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe: The KUKA-DLR lightweight robot arm – a new reference platform for robotics research and manufacturing, 41st Int. Symp. Robot. 6th Ger. Conf. Robot. (ROBOTIK), Munich (2010)

54.26 M. Hedelind, S. Kock: Requirements on flexible robot systems for small parts assembly: A case study, Proc. 2011 IEEE Int. Symp. Assem. Manuf. (ISAM), Tampere (2011) pp. 1–7

54.27 S. Kinkel, E. Kirner, H. Armbruster, A. Jäger: Relevance and innovation of production-related services in manufacturing industry, Int. J. Technol. Manag. **55**(3/4), 263–273 (2011)

54.28 International Organization for Standardization: ISO 8373. Robots and robotic devices – Vocabulary, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=53830 (2012)

54.29 R. Bloss: Review of manufacturing cells as they achieve high levels of autonomy and flexibility, Assem. Autom. **33**(2), 112–116 (2013)

54.30 A. Kahn: *The Encyclopedia of Work-related Illnesses, Injuries, and Health Issues* (Facts On File, New York 2004)

54.31 M. Wilson: Developments in robot applications for food manufacturing, Ind. Robot Int. J. **37**(6), 498–502 (2010)

54.32 R. Moreno Masey, J. Gray, T. Dodd, D. Caldwell: Guidelines for the design of low-cost robots for the food industry, Ind. Robot Int. J. **37**(6), 509–517 (2010)

54.33 K. Mathia: *Principles and Applications in Cleanroom Automation* (Cambridge Univ. Press, Cambridge 2010)

54.34 D. Rossi, E. Bertolini, M. Fenaroli, F. Marciano, M. Alberti: A multi-criteria ergonomic and performance methodology for evaluating alternatives in "manuable" material handling, Int. J. Ind. Ergon. **43**(4), 314–327 (2013)

54.35 D. Caldwell: *Robotics and Automation in the Food Industry: Current and Future Technologies* (Woodhead, Cambridge 2012)

54.36 M. Palzkill, A. Verl: Object pose detection in industrial environment, Proc. 7th Ger. Conf. Robot. (ROBOTIK), Munich (2012)

54.37 R. Mattone, M. Divona, A. Wolf: Sorting of items on a moving conveyor belt. Part 2: Performance evaluation and optimization of pick-and-place operations, Robot. Comput.-Integr. Manuf. **16**(2/3), 81–90 (2000)

54.38 A. Grzesiak, R. Becker, A. Verl: The bionic handling assistant: A success story of additive manufacturing, Assem. Autom. **4**, 329–333 (2011)

54.39 K. Ikeuchi, B. Horn, S. Nagata, T. Callahan, O. Fein: Picking up an object from a pile of objects, A.I. Memo No. 726 (Artif. Intell. Lab./MIT, Cambridge 1983)

54.40 M.-Y. Liu, O. Tuzel, A. Veeraraghavan, T. Marks, R. Chellappa: Fast object localization and pose estimation in heavy clutter for robotic bin picking, Int. J. Robotics Res. **31**(8), 951–973 (2012)

54.41 J.J. Rodrigues, J.-S. Kim, M. Furukawa, J. Xavier, P. Aguiar, T. Kanade: 6D pose estimation of textureless shiny objects using random ferns for bin picking, IEEE/RSJ Int. Conf. Intell. Robots Syst., Vilamoura (2012)

54.42 Z. Pan, J. Polden, N. Larkin, S. Van Duin, J. Norrish: Recent progress on programming methods for industrial robots, Robot. Comput.-Integr. Manuf. **28**(2), 87–94 (2012)

54.43 P. Klüger: Future body-in-white concepts. Increased sustainability with reduced investment and life-cycle-cost, Automot. Manuf. Solut., Shanghai (2013)

54.44 M. Shah, R.D. Eastman, T. Hong: An overview of robot-sensor calibration methods for evaluation of perception systems, Workshop Perform. Metr. Intell. Syst., New York (2012)

54.45 Deutsches Institut für Normung: DIN8593-0:2003-09: Manufacturing processes joining – Part 0: General; classification, subdivision, terms and definitions, http://www.beuth.de/de/norm/din-8593-0/65031206 (2003)

54.46 N. Lohse, H. Hirani, S. Ratchev, M. Turitto: An ontology for the definition and validation of assembly processes for evolvable assembly systems, Proc. 6th IEEE Int. Symp. Assem. Task Plan. (2005) pp. 242–247

54.47 K. Malecki: Vehicle body mounted door seals on a roll, Adhesion Adhesives Sealants **6**(1), 23–25 (2009)

54.48 S. Kock, T. Vittor, B. Matthias: Robot concept for scalable, flexible assembly automation, Proc. IEEE Int. Symp. Assem. Manuf. (ISAM), Tampere (2011)

54.49 H. Chen, J. Wang, G. Zhang, T. Fuhlbrigge, S. Kock: High-precision assembly automation based on robot compliance, Int. J. Adv. Manuf. Technol. **45**(9/10), 999–1006 (2009)

54.50 A. Stolt, M. Linderoth, A. Robertsson, R. Johansson: Force controlled robotic assembly without a force sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Saint Paul (2012)

54.51 ABB Robotics: Application manual – force control for machining, http://new.abb.com/products/robotics/application-equipment-and-accessories/flexfinishing/function-package-for-force-control (2013)

54.52 J. Domnick, Z. Yang, Q. Ye: Simulation of the film formation at a high-speed rotary bell atomizer used in automotive spray painting processes, Proc. 22nd Eur. Conf. Liq. At. Spray Syst. ILASS Eur., Como (2008)

54.53 M. Jonsson, A. Stolt, A. Robertsson, S. von Gegerfelt, K. Nilsson: On force control for assembly and deburring of castings, Prod. Eng. **7**(4), 351–360 (2013)

54.54 Z. Pan, H. Zhanh, Z. Zhu, J. Wang: Chatter analysis of robotic machining process, J. Mater. Process. Technol. **173**(3), 301–309 (2006)

54.55 M. Weck, C. Brecher: *Werkzeugmaschinen 2 – Konstruktion und Berechnung* (Springer, Berlin, Heidelberg 2006)

54.56 U. Schneider, M. Ansaloni, M. Drust, F. Leali, A. Verl: Experimental investigation of sources of error in robot machining, Int. Workshop Robot. Smart Manuf., Porto (2013)

54.57 J. Wang, H. Zhang, T. Fuhlbrigge: Improving machining accuracy with robot deformation compensation, Int. Conf. Intell. Robots Syst., St. Louis (2009)

54.58 C. Lehmann, B. Olofsson, K. Nilsson, M. Halbauer, M. Haage, A. Robertsson, O. Sornmo, U. Berger: Robot joint modeling and parameter identification using the clamping method, IFAC Conf. Manuf. Modell. Manag. Control, Saint Petersburg (2013)

54.59 B. Saund, R. DeVlieg: High accuracy articulated robots with CNC control system, SAE Int. J. Aerosp. **6**(2), 780–784 (2013)

54.60 U. Schneider, J.R. Diaz Posada, M. Drust, A. Verl: Position control of an industrial robot using an optical measurement system for machining purposes, Proc. 11th Int. Conf. Manuf. Res. (2013) pp. 307–312

54.61 U. Schneider, B. Olofsson, O. Sörnmo, M. Drust, A. Robertsson, M. Hägele, R. Johansson: Integrated approach to robotic machining with macro/micro actuation, Robotics Comput.-Integr. Manuf. **30**(6), 636–647 (2014)

54.62 International Organization for Standardization: ISO 12100:2010. Safety of machinery – General principles for design – Risk assessment and risk reduction, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=51528 (2010)

54.63 European Parliament: Directive 2006/42/EC of the European Parliament and the Council of 17 May 2006 on machinery, and amending Directive 95/16/EC (recast), http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2006:157:0024:0086:EN:PDF (2006)

54.64 Siemens: *Easy Implementation of the European Machinery Directive, Functional Safety of Machines and Systems* (Siemens, Fürth 2012)

54.65 International Organization for Standardization: ISO 13849-1:2006. Safety of machinery – Safety-related parts of control systems – Part 1: General principles for design, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=3491 (2006)

54.66 International Electrotechnical Commission: IEC 62061 ed1.0. Safety of machinery – Functional safety of safety-related electrical, electronic and programmable electronic control systems: http://webstore.iec.ch/webstore/webstore.nsf/artnum/033604!opendocument (2005)

54.67 International Electrotechnical Commission: IEC 61508 ed2.0. Functional safety of electrical/electronic/programmable electronic safety-related systems, http://www.iec.ch/functionalsafety/standards (2010)

54.68 International Organization for Standardization: ISO 13855:2010. Safety of machinery – Positioning of safeguards with respect to the approach speeds of parts of the human body, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=42845 (2010)

54.69 International Organization for Standardization: ISO 13857:2008. Safety of machinery – Safety distances to prevent hazard zones being reached by upper and lower limbs, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=39255 (2008)

54.70 R. Hollmann, A. Rost, M. Hägele, A. Verl: A HMM-based approach to learning probability models of programming strategies for industrial robots, 2010 IEEE Int. Conf. Robotics Autom., Anchorage (2010)

54.71 B. Lacevic, P. Rocco, A. Zanchettin: Safety assessment and control of robotic manipulators using danger field, IEEE Trans. Robotics **29**(5), 1257–1270 (2013)

54.72 International Organization for Standardization: ISO/TC 184/SC 2. Robots and robotic devices,

Avidbots Ex1016
Page 1443 of 2154

http://www.iso.org/iso/home/store/catalogue_tc/catalogue_tc_browse.htm?commid=54138&development=on (2014)

54.73 S. Oberer-Treitz, A. Puzik, A. Verl: Measuring the collision potential of industrial robots, 41st Int. Symp. Robot. 6th Germ. Conf. Robot. (2010)

54.74 R. Behrens, N. Elkmann: Study on meaningful and verified thresholds for minimizing the consequences of human-robot collisions, IEEE Int. Conf. Robotics Autom. (ICRA) (2014) pp. 3378–3383

54.75 S. Haddadin, S. Haddadin, A. Khoury, T. Rokahr, S. Parusel, R. Burgkart, A. Bicchi, A. Albu-Schaeffer: On making robots understand safety: Embedding injury knowledge into control, Int. J. Robotics Res. **31**(13), 1578–1602 (2012)

54.76 J. Marvel, J. Falco: *Best practices and performance metrics using force control for robot assembly* (Nat. Inst. Stand. Technol., Gaithersburg 2012)

54.77 R. Zurawski: *Integration Technologies for Industrial Automated Systems* (CRC, Boca Raton 2006)

Multimedia Contents

55. Space Robotics

Kazuya Yoshida, Brian Wilcox, Gerd Hirzinger, Roberto Lampariello

In the space community, any unmanned spacecraft can be called a robotic spacecraft. However, *Space Robots* are considered to be more capable devices that can facilitate manipulation, assembling, or servicing functions in orbit as assistants to astronauts, or to extend the areas and abilities of exploration on remote planets as surrogates for human explorers.

In this chapter, a concise digest of the historical overview and technical advances of two distinct types of space robotic systems, orbital robots and surface robots, is provided. In particular, Sect. 55.1 describes orbital robots, and Sect. 55.2 describes surface robots. In Sect. 55.3, the mathematical modeling of the dynamics and control using reference equations are discussed. Finally, advanced topics for future space exploration missions are addressed in Sect. 55.4.

55.1 **Historical Developments and Advances of Orbital Robotic Systems** 1424
55.1.1 Robotic Arms for Assistance of Human Space Flight 1424
55.1.2 Future-Oriented Space Robot Experiments 1426

55.2 **Historical Developments and Advances of Surface Robotic Systems** 1430
55.2.1 Teleoperated Rovers 1430
55.2.2 Autonomous Rovers 1432
55.2.3 Research Systems 1435
55.2.4 Sensing and Perception 1436
55.2.5 Estimation 1437
55.2.6 Manipulators for In-Situ Science 1437

55.3 **Mathematical Modeling** 1437
55.3.1 Space Robot as an Articulated Body System 1437
55.3.2 Equations for Free-Floating Manipulator Systems 1438
55.3.3 Generalized Jacobian and Inertia Matrices 1439
55.3.4 Linear and Angular Momenta 1440
55.3.5 Virtual Manipulator 1440
55.3.6 Dynamic Singularity 1441
55.3.7 Reaction Null-Space (RNS) 1441
55.3.8 Motion Planning Issues 1441
55.3.9 Equations for Flexible-Based Manipulator Systems 1443
55.3.10 Advanced Control for Flexible Structure Based Manipulators 1444
55.3.11 Contact Dynamics and Impedance Control 1446
55.3.12 Dynamics of Mobile Robots 1447
55.3.13 Wheel Traction Mechanics 1448

55.4 **Future Directions of Orbital and Surface Robotic Systems** 1452
55.4.1 Robotic Maintenance and Service Missions 1452
55.4.2 Robonaut and JUSTIN 1453
55.4.3 Aerial Platforms 1454
55.4.4 Mobility Concepts and Subsurface Platforms 1456

55.5 **Conclusions and Further Reading** 1457

Video-References 1457

References 1458

Avidbots Ex1016
Page 1445 of 2154

55.1 Historical Developments and Advances of Orbital Robotic Systems

Key issues in space robots and systems are characterized as follows:

Manipulation – Although manipulation is a basic technology in robotics, microgravity in the orbital environment requires special attention to the motion dynamics of manipulator arms and objects being handled. Reaction dynamics that affect the base body, impact dynamics when the robotic hand contacts an object to be handled, and vibration dynamics due to structural flexibility are included in this issue.

Mobility – The ability for locomotion is particularly important in exploration robots (rovers) that travel on the surface of a remote planet. These surfaces are natural and rough, and thus challenging to traverse. Sensing and perception, traction mechanics, and vehicle dynamics, control and navigation; all of these mobile robotics technologies must be demonstrated in a natural untouched environment.

Teleoperation and Autonomy – There is a significant time delay between a robotic system at a work site and a human operator in an operation room on the earth. In earlier orbital robotics demonstrations, the latency was typically 5 s, but can be several tens of minutes, or even hours for planetary missions. Telerobotics technology is therefore an indispensable ingredient in space robotics, and the introduction of autonomy is a reasonable consequence.

Extreme Environments – In addition to the microgravity environment that affects the manipulator dynamics or the natural and rough terrain that affects surface mobility, there are a number of issues related to extreme space environments that are challenging and must be solved in order to enable practical engineering applications. Such issues include extremely high or low temperatures, high vacuum or high pressure, corrosive atmospheres, ionizing radiation, and very fine dust.

The first robotic manipulator arm used in the orbital environment is the shuttle remote manipulator system (SRMS). It was successfully demonstrated in the STS-2 mission in 1981 and was operational until the end of the shuttle era. This success opened a new era of orbital robotics and inspired a number of mission concepts to the research community. One ultimate goal that has been discussed intensively after the early 1980s is the application to the rescue and servicing of malfunctioning spacecraft by a robotic free-flyer or free-flying space robot (e.g., ARAMIS report [55.1], Fig. 55.1). In later years, manned service missions were conducted for the capture-repair-deploy procedure of malfunctioning satellites (Intelsat 603 by STS-49, for example) and for the maintenance of the Hubble space telescope (STS-61, 82, 103, and 109). For all of the examples, the Space Shuttle, a manned spacecraft with dedicated maneuverability, was used. However, unmanned servicing missions have not yet become operational. Although there were several demonstration flights, such as ETS-VII and Orbital Express (to be elaborated later), the practical technologies for unmanned satellite servicing missions await solutions to future challenges.

55.1.1 Robotic Arms for Assistance of Human Space Flight

Space Shuttle Remote Manipulator System

Onboard the space shuttle, the SRMS, or Canadarm, is a mechanical arm that maneuvers a payload from the payload bay of the space shuttle orbiter to its deployment position and then releases it [55.2]. It can also grapple a free-flying payload, maneuver it to the payload bay of the orbiter, and berth it back into the orbiter. The SRMS was first used on the second Space Shuttle mission STS-2, launched in 1981. Since then, it was used more than 100 times during space shuttle flight missions, performing such payload deployment or berthing as well as assisting human extra vehicular activities (EVAs). Servicing and maintenance missions to the Hubble space telescope and construction tasks of the International Space Station (ISS) have also been

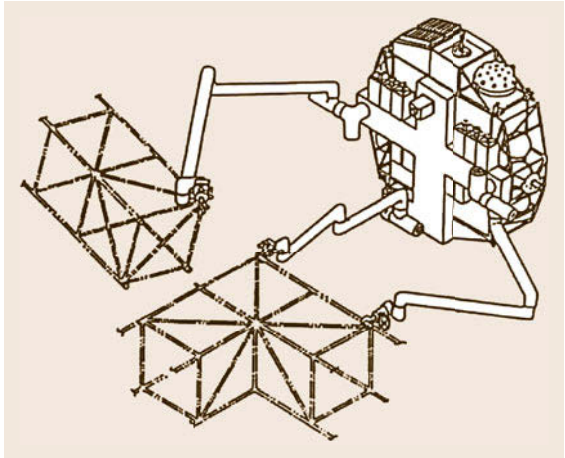


Fig. 55.1 A conceptual design of telerobotic servicer (after [55.1])

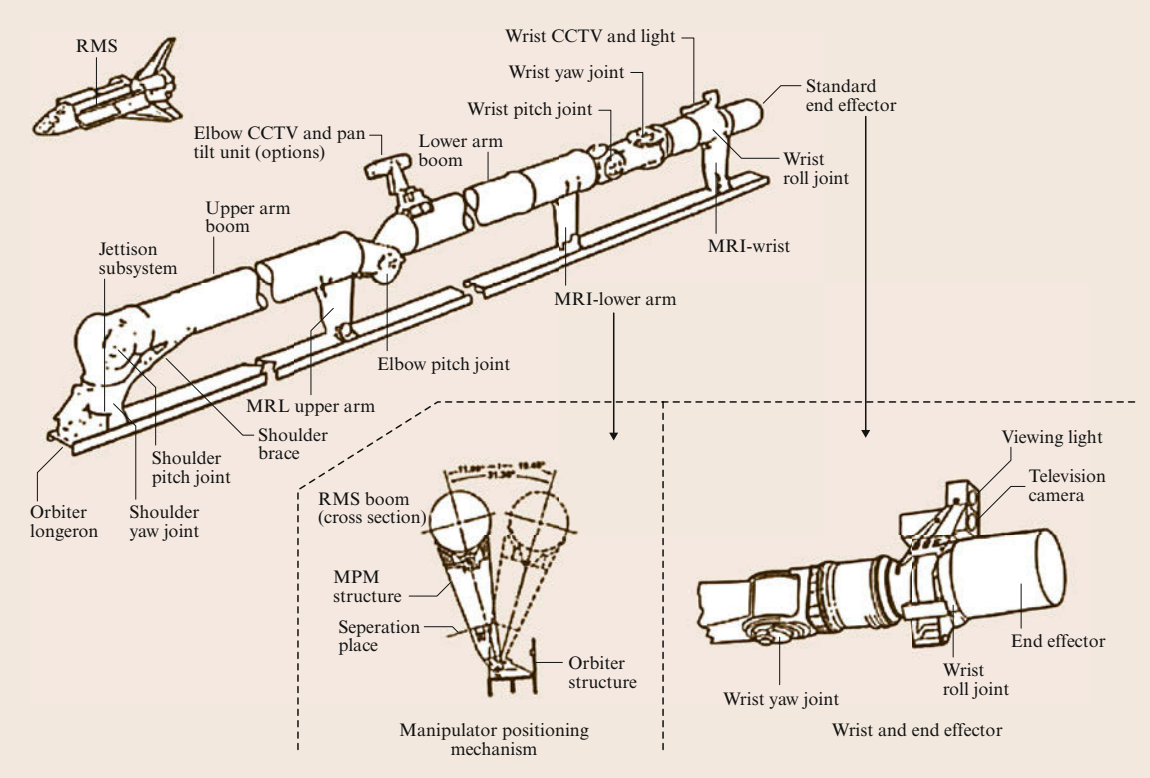


Fig. 55.2 Space Shuttle remote manipulator system (SRMS) (after [55.2])

successfully carried out by the cooperative use of the SRMS with human EVAs.

As depicted in Fig. 55.2, the SRMS arm was 15 m long and had 6-degrees of freedom (DOF), comprising shoulder yaw and pitch joints, an elbow pitch joint, and wrist pitch, yaw, and roll joints. Attached to the end of the arm was a special gripper system called the standard end effector (SEE), which was designed to grapple a pole-like fixture (GF) attached to the payload.

By attaching a foothold at the end point, the arm could serve as a mobile platform for an astronaut's EVAs (Fig. 55.3).

After the Space Shuttle COLUMBIA accident during STS-107, NASA outfitted the SRMS with the orbiter boom sensor system – a boom containing instruments to inspect the exterior of the shuttle for damage to the thermal protection system [55.3].

ISS Mounted Manipulator Systems

The ISS is the largest international technology project, with 15 countries making significant cooperative contributions. The ISS is an *outpost* of human presence in space, as well as a *flying laboratory* with substantial facilities for science and engineering research. In order



Fig. 55.3 Space shuttle remote manipulator system (SRMS) used as a platform for an astronaut's extra vehicular activity in the shuttle cargo bay

to facilitate various activities on the station, there are several robotic systems, some of which are already operational, while others are ready for launch.

The space station remote manipulator system (SSRMS), or Canadarm 2 (Fig. 55.4), was the next generation of SRMS for use on the ISS [55.4]. Launched in 2001 during STS-100 (ISS assembly flight 6A), the SSRMS has played a key role in the construction and maintenance of the ISS both by assisting astronauts during EVAs and using the SRMS on the Shuttle to hand over a payload from a Shuttle to the SSRMS. The arm is 17.6 m long when fully extended and has 7-DOF. Latching end effectors, through which power, data, and video can be transmitted to and from the arm, are attached to both ends. The SSRMS is self-relocatable using an inch-worm like movement with alternate grappling of power data grapple fixtures (PDGFs), which are installed over the station's exterior surfaces to provide the power, data, and video, as well as a foothold.

As another mobility aid for the SSRMS to cover wider areas of ISS, mobile base system (MBS) was added in 2002 by STS-111 (ISS assembly flight UF-2). The MBS provides lateral mobility as it traverses the rails on the main trusses [55.5].

The special purpose dexterous manipulator (SPDM), or Dextre, which is attached at the end of the SSRMS, is a capable mini-arm system to facilitate the delicate assembly tasks currently handled by astronauts during EVAs. The SPDM is a dual arm manipulator system, where each manipulator (with 3 m length) has 7-DOF and is mounted on a one degree-of-freedom body joint. Each arm has a special tool mechanism dedicated to the handling of standardized orbital replacement units (ORUs). The arms are teleoperated from a Robotic WorkStation (RWS) inside the space station [55.6].

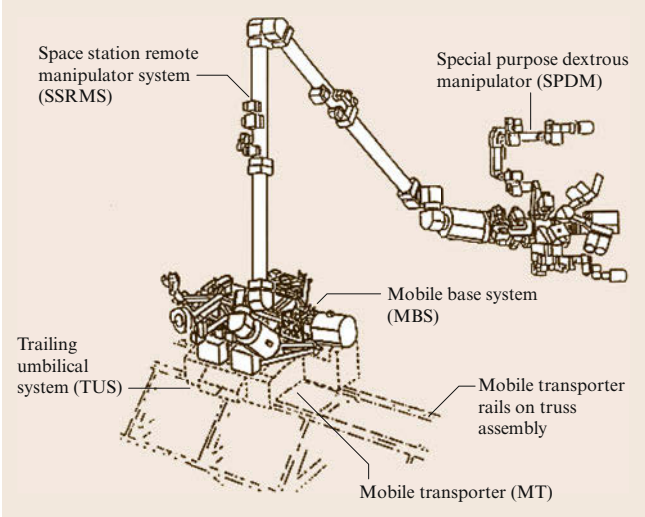


Fig. 55.4 Space station remote manipulator system (SSRMS) (after [55.4])

The European Space Agency (ESA) will also provide a robotic manipulator system for the ISS, the European robotic arm (ERA), and will be used mainly to work on the Russian segments of the station [55.7]. The arm is 11.3 m long and has 7-DOF. The basic configuration and functionality are similar to SSRMS [55.8].

In Japan, the Japanese experiment module remote manipulator system (JEMRMS), as shown in Fig. 55.5, was developed by the Japan Space Exploration Agency (JAXA) [55.9–11]. The arm was launched with the STS-124 Mission in 2008 and is now operative on the Japanese module of the ISS. JEMRMS comprises two components: the main arm, a 9.9 m long, six-degree-of-freedom arm, and the small fine arm, a 1.9 m long, six-degree-of-freedom arm.

Unlike the SSRMS or the ERA, the main arm does not have self-relocation capability, but is fitted with a small fine arm, with which JEMRMS can form a serial 12-degree-of-freedom macro-micro manipulator system. After installation, the arm is used to handle and relocate the components for the experiments and observations on the exposed facility.

55.1.2 Future-Oriented Space Robot Experiments

ROTEX

The Robot Technology Experiment (ROTEX) developed by the German Aerospace Agency (DLR), is one of the important milestones of robotics technology in space [55.12], as it demonstrated the first (remote) ground control of a space robot with of up to 6 s round-trip signal delay using geostationary relay satellites. A multisensory robotic arm was flown on Space Shuttle COLUMBIA (STS-55) in 1993. Although the robot worked inside a work cell on the shuttle, several key technologies, such as a multisensory gripper, teleoperation from the ground and by the astronauts, shared autonomy, and time-delay compensation by use of a predictive graphic display were successfully tested (Fig. 55.6 and VIDEO 330).

Presumably the most spectacular experiment in ROTEX was the fully automatic grasping of a small free-floating cube with flattened edges by the ground computers who evaluated the stereo images from the robot gripper, estimated the motion, predicted for the above-mentioned 6 s and sent up the commands for grasping (VIDEO 331).

Avidbots Ex1016
Page 1448 of 2154

ETS-VII

Engineering Test Satellite VII (ETS-VII), shown in Fig. 55.7, is another milestone in the development of robotics technology in space, particularly in the area of satellite servicing. ETS-VII was an unmanned spacecraft developed and launched by the National Space Development Agency of Japan (NASDA, currently JAXA) in November 1997. A number of experiments were successfully conducted using a 2 m long, six-degree-of-freedom manipulator arm mounted on its carrier satellite.

The mission objective of ETS-VII was to test free-flying robotics technology and to demonstrate its utility in unmanned orbital operation and servicing tasks. The mission consisted of two subtasks: autonomous rendezvous/docking (RVD) and a number of robot experiments (RBT). The robot experiments included: (1) teleoperation from the ground with a large time delay; (2) robotic servicing task demonstrations such as ORU exchange and deployment of a space structure; (3) dynamically coordinated control between the manipulator reaction and the satellite attitude response; and (4) capture and berthing of a cooperative target satellite [55.13].

The communication time delay due to radio propagation (speed-of-light) is relatively small, for example 0.25 s for a round trip to geostationary Earth orbit (GEO). However, to have a global coverage of communication in low Earth orbit (LEO) operations, the signals are transmitted via multiple nodes including data relay satellites located at GEO and ground stations. This makes the transmission distance longer, and even more additional delays are added at each node. As a result, the cumulative delay becomes some seconds, actually 5–7 s in case of ETS-VII mission, more or less the same as in the ROTEX experiment. However, it is important to state that these delays have been unnecessarily long as the optimal communication infrastructure was (and is still) missing. Figure 55.8 shows that for a low orbit robot satellite (with typically 1.5 h orbital period) half of the orbit (i. e., around 45 min) would have communication from the ground station via one relay satellite yielding approximately 600 ms round trip delay (including computational delays). For a robot in geostationary orbit having permanent communication to the ground station, the round trip delay would be only 300 ms.

In ETS-VII, opportunities for academic experiments were also opened to Japanese universities and European institutions (e.g., DLR and ESA), and important flight data were obtained that validate the concepts and theories for free-flying space robots [55.14, 15]. As an example, DLR performed experiments aiming at demonstrating robot control methods for performing ORU exchange tasks (see for example VIDEO 332) and dedicated satellite attitude control via *swimming* or *waiving* motions of the robot arm.

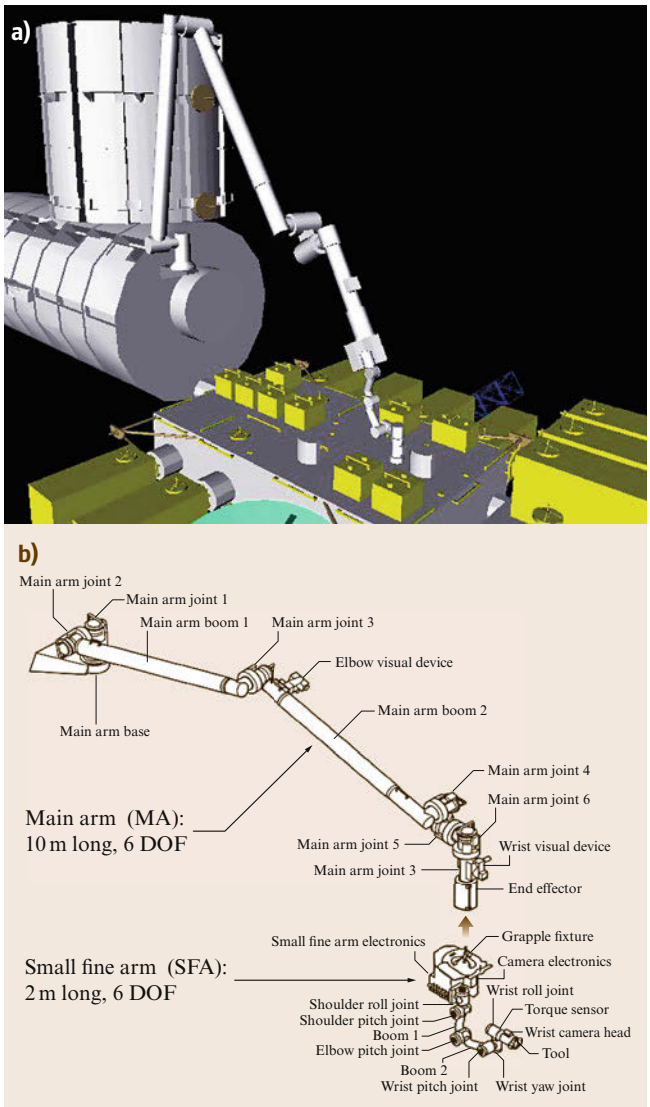


Fig. 55.5 (a) Japan Experiment Module (JEM) on the ISS and (b) the JEMRMS manipulator system

Ranger

Ranger is a teleoperated space robot being developed at the University of Maryland, Space Systems Laboratory [55.16]. Ranger consists of two seven-degree-of-freedom manipulators with interchangeable end effectors to perform such tasks as changeout of orbital replacement units (ORUs) in orbit. Also discussed was the changeout of the electronics controller unit (ECU) of the Hubble space telescope, which previously required human EVA. A number of tests and demon-

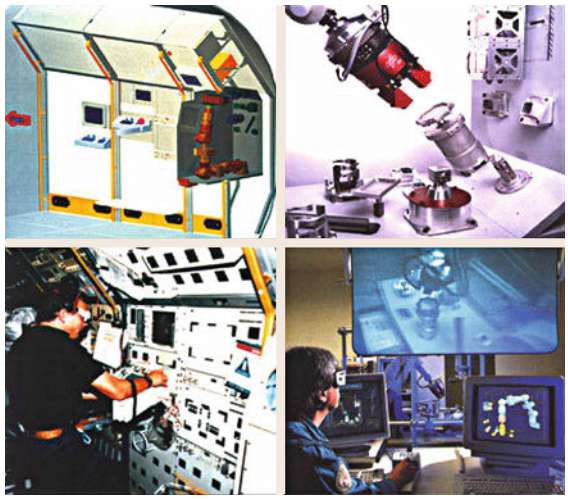


Fig. 55.6 ROTEX manipulator arm onboard the Spacelab D2 mission, the first remotely controlled space robot

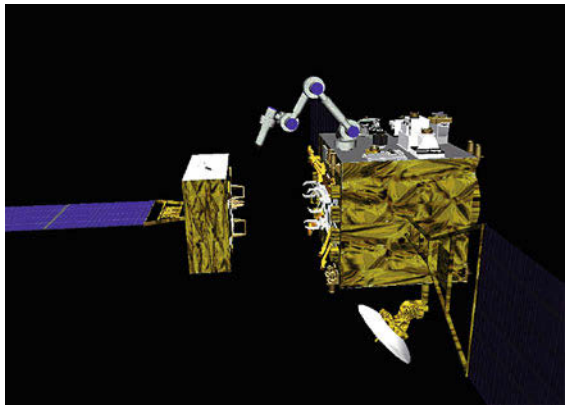


Fig. 55.7 Japanese Engineering Test Satellite ETS-VII

strations for servicing missions have been conducted at the University of Maryland Neutral Buoyancy Facility (Fig. 55.9). Originally designed for a free-flying flight experiment, Ranger had been redesigned for a shuttle flight experiment, but ultimately was not manifested on a shuttle flight.

Orbital Express

The Orbital Express Space Operations Architecture program is a DARPA program developed to validate the technical feasibility of robotic on-orbit refueling and reconfiguration of satellites, as well as autonomous rendezvous, docking, and manipulator berthing [55.17]. The system consists of the autonomous space transport robotic operations (ASTRO) vehicle, developed by Boeing Integrated Defense Systems, and a prototype modular next-generation serviceable satellite, NextSat, developed by Ball Aerospace. The ASTRO vehicle is

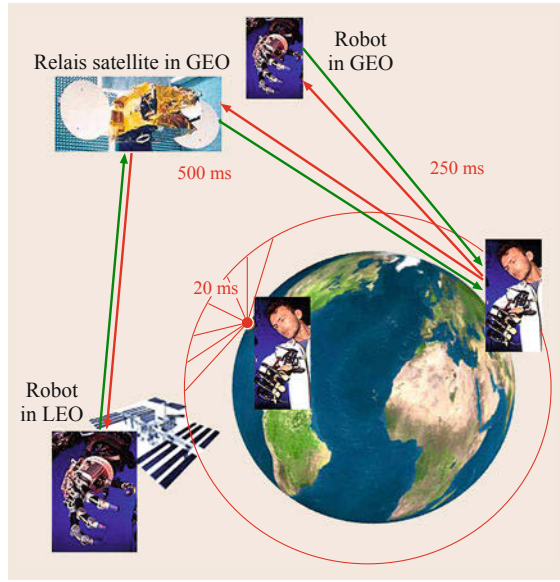


Fig. 55.8 Minimal *Roundtrip* signal propagation delay to robots in LEO is around 0.6 s, to robots in GEO around 0.3 s

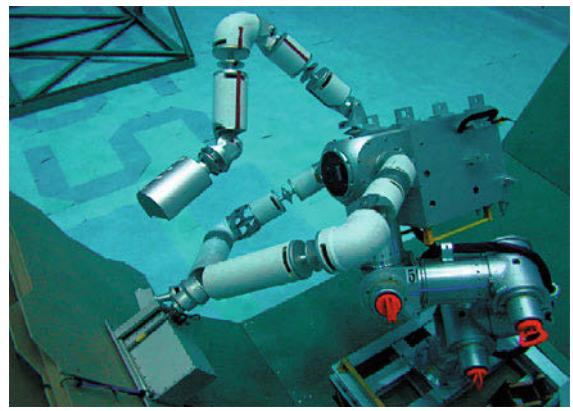


Fig. 55.9 Neutral buoyancy test of the Ranger telerobotic shuttle experiment

equipped with a robotic arm to perform satellite capture and ORU exchange operations (Fig. 55.10).

After its launch in March 2007, various mission scenarios were conducted. These scenarios include (1) visual inspection, fuel transfer, and ORU exchange on NextSat using ASTRO's manipulator arm when both spacecrafts are connected, (2) separation of NextSat from ASTRO, orbital maneuvers by ASTRO, and fly-around, rendezvous, and docking with NextSat, and (3) capture of NextSat using ASTRO's manipulator arm.

These scenarios were successfully completed by July 2007, with ASTRO's onboard autonomy using onboard cameras and advanced video guidance system.

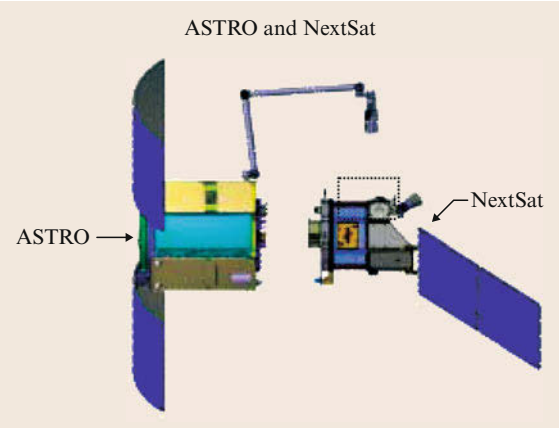


Fig. 55.10 Orbital express flight mission configuration

ROKVISS and Delayed Teleoperation

As a precursor demonstration of the planned German Orbital Servicing demonstration mission DEOS (see below), the German Aerospace Agency (DLR) has developed and flown a 0.5 m long, two-degree-of-freedom manipulator arm with a dedicated test bench, called robotic components verification on the ISS (ROKVISS) (Fig. 55.11) [55.18].

Using the 7–8 min contact window of the ISS overflight through a dedicated real-time space link, resulting in a roundtrip delay as small as 20 ms, this was the first high-fidelity telepresence system in the history of space flight that allowed force feedback teleoperation. A large amount of data documenting the evolution of electrical and mechanical properties of the robot (such as sensor accuracy, friction, and motor parameters) has been collected during the six year mission.

ROKVISS was launched by an unmanned Russian progress transport vehicle in 2004 and installed on the outer platform of the Russian segment of the station in early 2005 (VIDEO 333).

After six years, the system was still fully functional, even though it had undergone only low-cost qualification, i. e., most of the electronic components were off-the-shelf and the system as a whole had been qualified by radiation, vibration, and temperature tests. The long-time verification of these joints for outer space has been a second main goal of ROKVISS (VIDEO 334). Toward the end of the mission, ROKVISS was teleoperated from the private home of the project leader using standard internet communication to DLR's professional ground station. Round-trip delay went then up to around 400 ms still allowing telepresence with force-reflection. At the end of 2011, ROKVISS was brought back to Earth for completion of verification and validation tests (VIDEO 336).

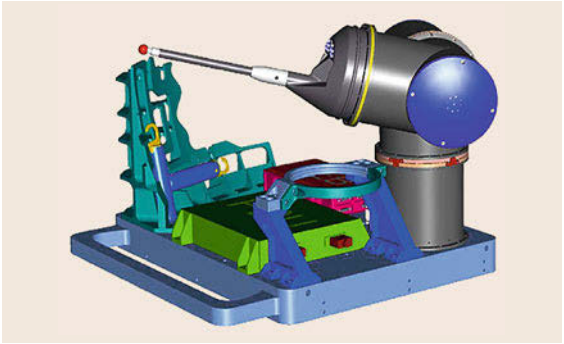


Fig. 55.11 ROKVISS – 6 years on ISS

Although the numbers of joints was small, challenging experiments of telepresence were conducted, in which human operators from ground teleoperated the arm using a force feedback joystick and stereo vision images. The ROKVISS arm, on the other hand, was equipped with joint torque sensors, allowing full impedance control and advanced bilateral control techniques. The secondary goal of the experiment was the space qualification of the joint drives, which are the key components of DLR's torque-controlled lightweight robots [55.19].

No electrical or mechanical damages were visible, and the main observation was that joint friction had nearly doubled in vacuum on ISS, but went back to its original value when the arm was back on earth again. In the framework of ROKVISS and the accompanying investigations, important basic clarifications of the general telepresence potentials and prerequisites could be achieved.

Already in the early days of space telerobotics as pushed forward by JPL, there have been estimates that humans might be able to master signal delays of up to 1 s in the visual system (i. e., looking at delayed images) and up to 500 ms with the haptic system from the proprioceptive point of view.

The interesting, fairly recent result is here [55.20, 21], where feasibility of force-reflecting teleoperation with communication delays as high as 650 ms has been proven. By using the so-called time domain passivity control approach, the mechanical energy of the system is observed and controlled in real time, such that the system is passive for any given communication channel characteristics, including varying time delays and packet loss. These results have been verified with a unique teleperesence experiment, where two torque-controlled light weight arms located in Oberpfaffenhofen (Germany) were connected through a real relayed communication link that used the geostationary satellite ARTEMIS, ground station communication an-

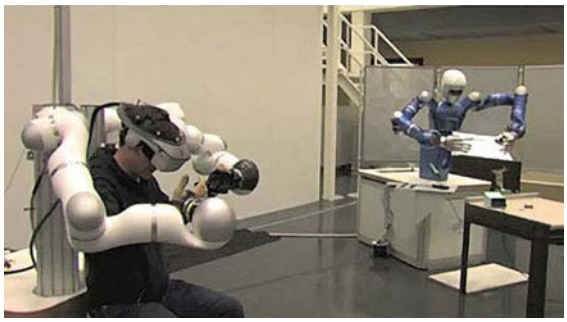


Fig. 55.12 Force-reflecting telepresence demos at DLR

tennas located in Garching (Germany) and a data mirror in Redu (Belgium) [55.22].

Recently, telepresence experiments with a fairly complex multidegree-of-freedom (DOF) system have been performed [55.23]. The system consisted of the DLR's *Space JUSTIN* humanoid robot and a human–machine interface based on two torque-controlled light weight arms as a force reflecting hand controller (haptic device) (Fig. 55.12 and VIDEO 337). Complex tasks that required with high levels of dexterity, such (un)screwing with a screwdriver and soldering, were performed with time delays up to 500 ms (VIDEO 338).

It is therefore concluded that teleoperation with force feedback is possible over the whole earth orbit.

The DEOS Demonstration Project

The German space robotics-demonstration mission DEOS, based on the ROKVISS-joint torque-controlled arm technology, aims at the demonstration of the maturity and availability of key technologies as needed for an orbit servicing. DEOS is comprised of two satellites, servicer and client (Fig. 55.13). Its main goal is to find, approach, and capture an uncontrolled and uncooperative satellite in LEO (VIDEO 339). After successful capturing some typical repair and maintenance tasks are planned to be demonstrated before the servicing satellite enters the atmosphere for a controlled descent together with the captured satellite. Project developments are currently being pursued to advance the technological readiness level, in view of future missions. Developments also include adaptations to Active Debris Removal missions in study, such as e.Deorbit (ESA), for the deorbiting of the ENVISAT satellite.



Fig. 55.13 DEOS – orbital servicing experiment with client (*left*) and servicer (*right*)

Robonaut and JUSTIN

Humanoid (human-like) robots have been developed to conduct human-compatible dexterous tasks. NASA's Robonaut and DLR's JUSTIN are such representative examples. These robots are elaborated in Sect. 55.4.2.

55.2 Historical Developments and Advances of Surface Robotic Systems

The research on surface exploration rovers began in the mid-1960s, with an initiative (that never flew) for an unmanned rover for the Surveyor lunar landers and a manned rover (Moon buggy) for the human landers in the United States. In the same period, research and development began for a teleoperated rover named Lunokhod in the Soviet Union. Both the Apollo-manned rover and the Lunokhod-unmanned rover were successfully demonstrated in the early 1970s on Moon [55.24]. In the 1990s, the exploration target had expanded to Mars, and in 1997, the Mars Pathfinder Mission successfully deployed a microrover named Sojourner that safely traversed the rocky field adjacent to the landing site by autonomously avoiding obstacles [55.25, 26]. Following this success, today, autonomous robotic vehicles are considered indispensable technology for planetary exploration. The twin Mars Exploration Rovers, Sprit and Opportunity, were launched in 2003, and have had remarkable success in terms of remaining operational in the harsh environment of Mars for over three years. Each has traveled more than 5000 m and has made significant scientific discoveries using on-board instruments [55.27, 28].

55.2.1 Teleoperated Rovers

The first remotely operated robotic space surface vehicle was Lunokhod (Fig. 55.14) [55.24]. Lunokhod 1 landed on the Moon on November 17, 1970 as a payload on the lander Luna-17, and Lunokhod 2 landed on the



Fig. 55.14 Lunokhod

Moon on January 16, 1973. Both were 8-wheeled skid-steered vehicles having a mass of about 840 kg, where almost all the components were in a pressurized *bathtub* thermal enclosure with a lid that closed over the tub to allow it to survive the deep cold ($\approx$ 100 K) of the long lunar nights using only the heat emitted by small pellets of radioisotope. On the inside of the lid were solar arrays which recharged batteries during the day as required to maintain operation of the vehicle. Lunokhod 1 operated for 322 Earth days, traversing over 10.5 km during that period, and returned over 20 000 TV images, 200 high-resolution panoramas, and the results of more than 500 soil penetrometer tests and 25 soil analyses using its x-ray fluorescence spectrometer. Lunokhod 2 operated for about 4 months, having traversed more than 37 km, with the mission officially terminated on June 4, 1973. It has been reported that Lunokhod 2 was lost prematurely when it began sliding down a crater slope and hasty commands were sent in response which ultimately caused end-of-mission.

Each of the eight wheels on the Lunokhod vehicles were 0.51 m in diameter and 0.2 m wide, giving an effective ground pressure of less than 5 kPa based on an assumed sinkage of 3 cm. Each wheel had a brush-type DC motor, a planetary gear reduction, a brake, and a dis-engagement mechanism allowing it to free-wheel in the event of some problem with the motor or gears. The mobility commands to the vehicle included two speeds forward or backward, braking, and turning to the right or the left either while moving or in place.

The vehicles had both gyroscope and accelerometer-based tilt sensors which could automatically stop the vehicle in the event of excessive tilt of the chassis. Typical mobility commands specified a time duration over which the motors would run, and then stop. Precision turning commands specified the angle through which the vehicle should turn. These commands were terminated when the specified turn angle had been reached according to the heading gyroscope. Odometry was determined by a ninth *small wheel* which was unpowered and lightly loaded and used only to determine over-the-ground distance. There was an on-board current overload system, and motor currents, pitch and roll measurements, distance traversed, and many component temperatures were telemetered to the ground operators.

The Lunakod crew consisted of a driver, a navigator, a lead engineer, an operator for steering the *pencil-beam* communication antenna, and a crew commander. The driver viewed a monoscopic television image from the vehicle, and gave the appropriate commands (*turn*, *proceed*, *stop*, or *back up*) along with their associated parametric value in terms of duration or angle. The navigator viewed displays of telemetry from the vehicles' *course gyroscope*, *gyrovertical sensor*, and odometer, and was responsible for calculating the trajectory of the vehicle and laying out the route to be followed. Thus, the driver was responsible for vehicle stability about its center of mass and the navigator was responsible for the trajectory of that center of mass. The lead engineer (assisted by many specialists as required), was responsible for assessing the health of the on-board systems. The lead engineer provided both routine updates on energy supply, thermal conditions, etc., as well as possible emergency alerts such as extreme motor currents or chassis tilt. The pencil-beam antenna operator oversaw the functioning of an independent ground-based closed-loop control system that servoed the antenna to always point at Earth, independent of the vehicle motion. The crew commander supervised the implementation and execution of the overall plan, gave any detailed commands for making actual contact with the surface (e.g., by the penetrometer) and also could override any command to the vehicle as he viewed the same information as the driver.

The entire driving system was tested extensively prior to the Lunokhod 1 mission at a *lunodrome* having simulated lunar terrain which proved to be more challenging than that actually encountered during the Lunokhod 1 mission. Despite this, the operators of Lunokhod 1 said they *encountered a dangerous situ-*

Avidbots Ex1016
Page 1453 of 2154

ation (unforeseen entrance into a crater, rolling onto a rock, etc.) slightly more often than once per kilometer. This was attributed to inadequate driving experience, the modest quality of the television images, and the poor illumination conditions on the Moon. The driving direction was often selected primarily to give the best images; even so, the operators reported *fictitious dangers* caused by varying illumination conditions. In the first three months (lunar days) of operation, the vehicle traversed 5224 m in 49 h of driving using 1695 driving commands, including about 500 turns. Sixteen signals were sent for protection against excessive tilt during that time; approximately 140 craters were traversed at maximum slope angles of 30°.

With the approximately 2.6 s speed-of-light delay, the operators stated that *control experience confirmed the desirability of movement in a starting-stopping regime with mandatory stopping each few meters.* The soil properties were found to *differ substantially even in terrain sectors not very distant from one another.* The soil penetrometer determined that the upper layer of regolith varied from a stiffest where the penetrometer required about 16 kg (Earth weight) of force to penetrate about 26 mm, to a weakest measurement where only 3 kg of Earth weight caused a penetration of about 39 mm. The cone penetrometer had a base diameter of 50 mm and a cone height of 44 mm. Thus the upper layer of regolith had a rate of increase of loadbearing strength ranging from about 400 kPa m^{-1} for the weakest soil to about 3 MPa m^{-1} for the stiffest soil. Crater walls and the immediate ejecta blanket around craters generally exhibited the weakest soil. Below 5–10 cm of penetration depth, the regolith generally became rapidly stiffer. The mean value of wheel slippage for the first three lunar days was about 10%. On horizontal terrain, the slippage ranged from zero to 15% depending on the surface irregularities and ground inhomogeneity. On crater slopes, the slip increased to 20–30%. The specific resistance of the Lunokhod wheels was generally in the range of 0.05–0.25, while the specific free traction (the ratio of traction to weight) was in the range of 0.2–0.41. The crater distribution in the area explored by Lunokhod 1 was found to be closely approximated by the formula $N(D) = AD^{-\delta}$, where $N(D)$ is the number of craters larger than diameter D meters per hectare of lunar surface, A is a scale factor found to be about 250, and δ is the distribution exponent, found to be about 1.4 [55.24].

55.2.2 Autonomous Rovers

In the mid-1960s, research began on a lunar rover at the US Jet Propulsion Laboratory (JPL) in Pasadena, California, when it was proposed to put a small rover on the Surveyor lunar landers. These landers were led by JPL (based on a system-level contract to Hughes), and were designed to land softly on the Moon to establish the safety of such landing prior to the Apollo landers with humans aboard. At the time it was speculated (notably by T. Gold) that the Moon might be covered in a thick layer of soft dust that would *swallow* any lander. In 1963, JPL issued a contract to build a small rover concept prototype in support of the Surveyor program to the General Motors Defense Research Laboratories in Goleta, California. That GM facility had recently hired *Bekker*, who was considered *the father of off-road locomotion*, having written several seminal textbooks on the subject, and having introduced many of the key concepts relating soil properties to off-road vehicle performance that are still used today [55.29, 30] (Sect. 55.3.12).

Bekker and his team proposed an articulated 6-wheeled vehicle based on a novel 3 cab configuration with an axial spring-steel suspension. This vehicle exhibited remarkable mobility, being able to climb vertical steps up to 3 wheel radii high, and crossing crevasses 3 wheel radii wide. Notable people working with Bekker were Farenc Pavlics, who went on to lead the development of the mobility system for the Apollo lunar rover (under contract from Boeing), and Fred Jindra, who developed the underlying equations describing the mobility of the 6-wheeled articulated vehicle that were later used by Don Bickler in conceiving of the *rocker bogie* chassis used on Sojourner and the Mars Exploration Rovers. Bekker and his team proposed the 6 wheeled articulated vehicle after experimenting with many types of vehicles, including multitracked vehicles, screw-type vehicles (for fine powdered terrain), and others. The 6-wheeled vehicle demonstrated in scale-model testing superior performance in both soft and rocky terrain.

They built and delivered two vehicles that were about 2 m long with approximately 0.5 m wheel diameters. Those vehicles were used in testing throughout the 1960s and early 1970s to conduct simulated operations to determine how such vehicles could actually be used on the Moon. One key issue was that the speed-of-light round trip from the Moon (about 3 s) precluded direct *driving* of the vehicle. Perhaps most annoying was the fact that, during vehicle motion, the highly directional radio antenna used to communicate with Earth would lose its pointing, and so communications would briefly be lost. This meant that operators driving the rover would be confronted with a series of still images, instead of a stream of moving images. It was quickly realized that much of an operator's *situation awareness* and depth perception needed to drive a vehicle with a monocular camera comes from motion. It

was very difficult to drive from frozen monocular images. A crude form of stereo was incorporated where the camera mast was raised and lowered slightly and the operator could switch back and forth between the two views.

Following the successful landing of several of the Surveyor spacecraft, and the discovery that all landing sites seemed to have relatively firm soil, it was concluded that the Surveyor lunar rover was not needed. As a result, the prototype was used for research into the early 1970s, and subsequently restored for use again in research in the 1980s, becoming the first vehicle to be outfitted with *waypoint navigation* of the sort later used on the Sojourner and MER missions.

About the time Viking was conceived and developed, JPL began the *1984* Mars rover effort. (1984 was an energetically favorable launch opportunity from Earth to Mars, and the next likely major mission opportunity after Viking.) Two testbed vehicles were developed, a *software* prototype and a *hardware* prototype. The software prototype had a *Stanford* arm, designed by Vic Scheinman (who went on to design the Unimation PUMA arm and many other famous early robotic devices). This was the only *1.5-scale* Stanford arm ever built. *Lewis* and *Bejczy* became well-known in robotics for solving the kinematics of this arm, one of is not the first full kinematics ever done in robotics up to that time, e.g., [55.31]. A stereo pan-tilt head was implemented and equipped with the first solid-state cameras to become available. A number of very important works were published in the 1977 International Joint Conference on artificial intelligence, e.g., [55.32, 33]. The first *hand-eye-locomotion* coordination was done with this vehicle, where a rock was designated in a stereo image, and the vehicle maneuvered autonomously to a point where the arm could reach out and pick up the rock. One of the first demonstrations of *pin in hole* insertion and other *dexterous* manipulations were also done with this system in the 1970s.

The *hardware* prototype was built using elastic *loop wheels* made by Lockheed [55.34]. The vehicle was battery powered and controlled via a handheld RC unit of the type used by hobbyists.

In late 1982, JPL had a contract with the US Army to study the use of robotic vehicles in support of the US Army. During this study, Brian Wilcox at JPL proposed a technique to reduce the need for a real-time video link or high-bandwidth communication channel between the vehicle and the operator. This technique (which became known as computer-aided remote driving, or CARD) [55.35] required the transmission of a single stereo image from the vehicle to the operator, so the operator could designate waypoints in that image using a 3-D cursor. By use of a single stereo image instead of a continuous stream of monocular images, the amount of information that needed to be transmitted by the vehicle was reduced by orders of magnitude. JPL first demonstrated CARD on the resurrected surveyor lunar rover vehicle (SLRV, which had been painted baby-blue and so became known as the *Blue Rover*, Fig. 55.15a), and later on a modified Humvee. During field tests in the Mojave desert in 1988, CARD was demonstrated on the Humvee with path designations of 100 m per stereo image, and with time-to-designate each path of only a few seconds.

As the CARD work was ongoing, an internally funded effort at JPL demonstrated a concept called *semiautonomous navigation* (SAN). This concept involved humans on Earth designating *global* paths using maps of the sort that could be developed from orbiter imagery, and then having the vehicle autonomously refine and execute a *local* path that avoids hazards. The moderate success of that effort led to a NASA funded effort, leading to the development of a new vehicle, called Robby (Fig. 55.15b). Robby was a larger vehicle that could support the on-board computing and power

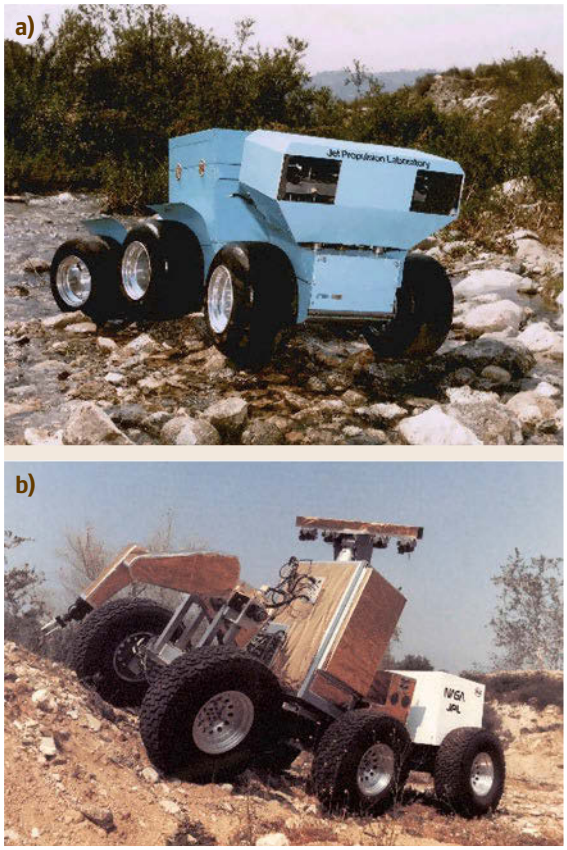


Fig.55.15a,b Six-wheel articulated body rovers developed by JPL **(a)** SLRV and **(b)** Robby

needed for untethered operation. (The SLRV had been tethered to a VAX 11/750 minicomputer over a 1500 foot tether during arroyo field testing of CARD.) For the first time (in 1990), an autonomous vehicle had made a traverse through an obstacle field that was faster than a rover could have done on Mars using human path designation done on Earth.

However, Robby had a severe *public relations* problem – it was perceived as too large. Of course, none of the computers or power systems had been miniaturized or lightweighted – it was composed entirely of the lowest-cost components that could do the job. However, because it was the size of a large automobile, observers and NASA management got the impression that future rovers would be car-sized or even truck-sized vehicles. This was compounded by the Mars rover sample return (MRSR) study done by JPL in the late-1980s which suggested a mass for the rover of 882 kg. An independent study of the MRSR study by Science Applications International, Inc. (SAIC) estimated the overall cost at $13B. When word of this *outrageous* price tag filtered around NASA Headquarters and into the Congressional Staff, MRSR was summarily killed. Robby died along with it. At about the same time, NASA funded Carnegie Mellon University to develop *Ambler* (Fig. 55.16), a large walking robot that was able to autonomously choose safe footfall locations, also as a testbed Mars Rover [55.36, 37]. Ambler had a similar public-relations problem, being about the same mass as Robby, that the NASA management community was very skeptical that such large systems could affordably be flown to Mars. Both Robby and Ambler had all-onboard power and computing systems, which at that time were not sufficiently miniaturized to make autonomous rovers credible for actual flight missions. *Moore's law* was not only causing the computing technology to become miniaturized at a high rate, but also the energy required per computing instruction was dropping rapidly. This meant that early systems devoted most of their power to computing rather than to motive power. Later systems, such as the Mars exploration rovers, have a more nearly equal balance between power for mobility and power for computation. Future systems will presumably devote the majority of their power to mobility as opposed to computation.

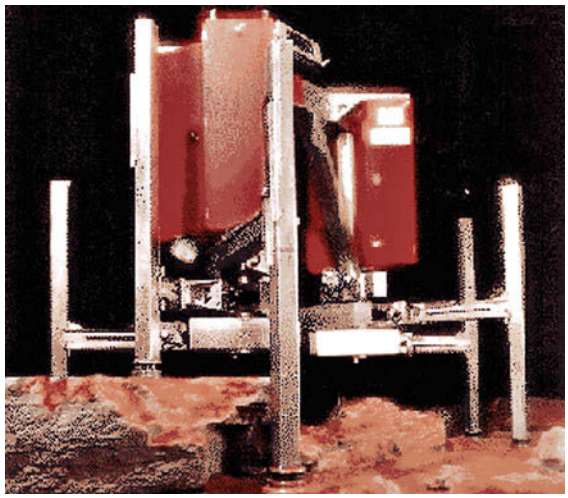


Fig. 55.16 Ambler

Soon thereafter, the Mars Environmental Survey (MESUR) mission set was proposed, as a lower cost alternative to a sample return mission. The MESUR Pathfinder mission was proposed as a first test of what was envisioned as a network of 16–20 surface stations to provide global coverage of Mars. A small rover was proposed to the Mars Science Working group [55.38, 39]. A very short-term development effort culminated in a demonstration in July 1992 of a ≈ 4 kg rover that could move to directed points on the surface nearby a lander using stereo designation of waypoints in a 3-D display of frozen images taken from a lander mast camera pair (Fig. 55.17). This demonstration was sufficiently successful that a similar rover was manifested for the Mars Pathfinder mission. The Pathfinder rover (Fig. 55.18) was later named Sojourner, and became the first autonomous vehicle to traverse the surface of another planet, using a hazard detection and avoidance system to move safely between waypoints through a rockfield [55.25, 26]. The hazard detection system avoided obstacles, and also was used to position the vehicle accurately in front of rocks. Sojourner operated successfully for 83 Mars days (until the failure of the lander, which was acting as a communications relay between the rover and Earth). Sojourner examined approximately a dozen rock and soil samples with its Alpha-Proton-x-ray spectrometer, which gives the elemental composition of the rocks and soil. The success of Sojourner led directly to the decision to build the twin Mars exploration rovers launched in 2003. Both Sojourner and the subsequent Mars exploration rovers *Spirit* and *Opportunity*, and the Mars Science Laboratory *Curiosity*, use waypoint designation in stereo images by the human operator together with autonomous hazard detection and avoidance to keep the rovers safe if they should wander off the designated path.

During the 1992–1993 summer season in Antarctica, the Dante I robot, built by Carnegie-Mellon University and funded by NASA, attempted to rappel into the caldera of the active volcano Mt. Erebus. Dante was a walking robot, and was the first serious attempt to make a robot rappel down a grade that was too steep to traverse using purely frictional contact. Unfortunately, the extreme cold (even in the summer) compounded

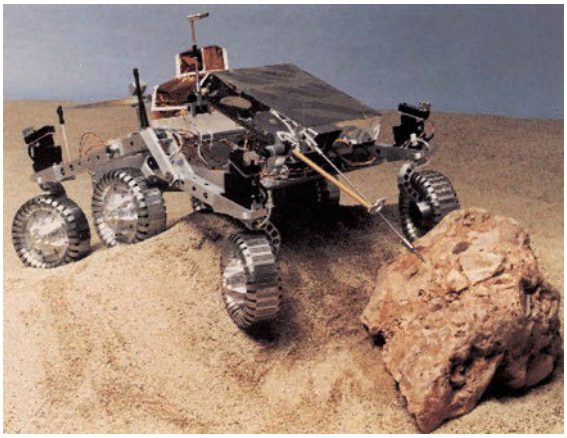


Fig. 55.17 Rocky 4

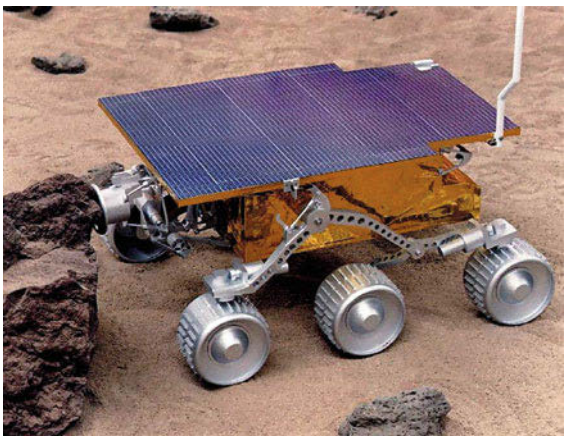


Fig. 55.18 The Pathfinder rover, Sojourner

by human error caused a kink in the fiber-optic umbilical to snag going through an eyelet, breaking the high-bandwidth fiber communications on which the system depended. The fiber could not be repaired in the field, and so the mission was aborted. Undaunted, in the summer of 1994, Dante II (Fig. 55.19) made a successful rappel into the caldera of Mt. Spur in Alaska, exploring the active vents on the crater floor in a way that would be unsafe in the extreme if done with human explorers. The Dante robot series demonstrated that rappelling, especially when combined with legged locomotion, allows robots to conduct exploration to extremely hazardous sites in ways that humans cannot.

In 1984, NASA started the Telerobotics Research program [55.40, 41]. This program demonstrated various aspects of on-orbit assembly, maintenance, and servicing. Some highlights of this activity were the automated tracking and grappling of a free-spinning satellite (suspended with a counterweight and gimbal

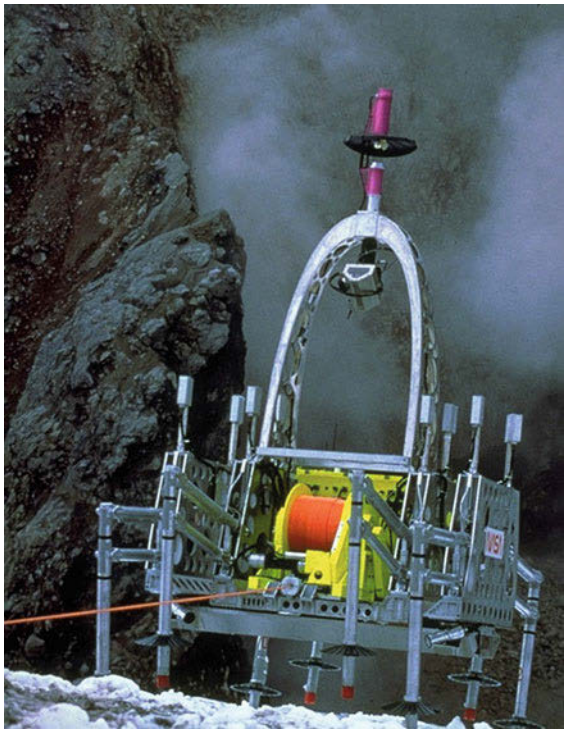


Fig. 55.19 Dante II at Mt. Spur in Alaska

for realistic reactions under external forces), connection of a flight-like fluid coupler, and many *busy box* functions such as door opening, threaded fastener mating and demating, use of power tools, dual-arm manipulation of a simulated hatch cover and flexible thermal blanket, etc., by various control approaches ranging from force-reflecting teleoperation to fully autonomous sequences. This activity ended in about 1990.

55.2.3 Research Systems

There have been many mobile robots built by government, university, and industrial groups whose objective was to develop new technologies for planetary surface exploration, or to excite students or young engineers about the possibilities in that area. Carnegie Mellon University developed the Ambler, Dante, Nomad, Hyperion, Zoe, and Icebreaker robot series. The Jet Propulsion Laboratory (JPL), Draper Labs, MIT, Sandia National Lab, and Martin-Marietta (later Lockheed-Martin) each built more than one planetary surface robot testbed. The Marsokhod chassis built by VNII Transmach of St. Petersburg, Russia was used by research groups there and also in Toulouse (LAAS and CNRS) [55.42, 43] as well as the NASA Ames Research Center and McDonnell Douglas Corporation (later part of the Boeing Company) in the US.

These research platforms have been used for two basic avenues of research. One avenue is to perfect safe driving techniques on planetary surfaces, despite the speed-of-light latencies inherent in robotic exploration of the planets. This includes the waypoint navigation technology developed at JPL in the 1980s, where frozen stereo images are used to plan a possibly lengthy series of waypoints or activity sites, and then executed with various sorts of reflexive hazard avoidance or safing techniques, such as used on the Sojourner rover on Mars in 1997. The other is to develop higher level autonomy for improved science data return or mission robustness. Technologies in this latter category include mission planners that attempt to optimize routes and activity sequences based on time, limits to peak power, total energy, expected temperature, illumination angles, availability of communications, and others.



Fig. 55.20 The Mars exploration rovers, Spirit and Opportunity, with a manipulator arm in front

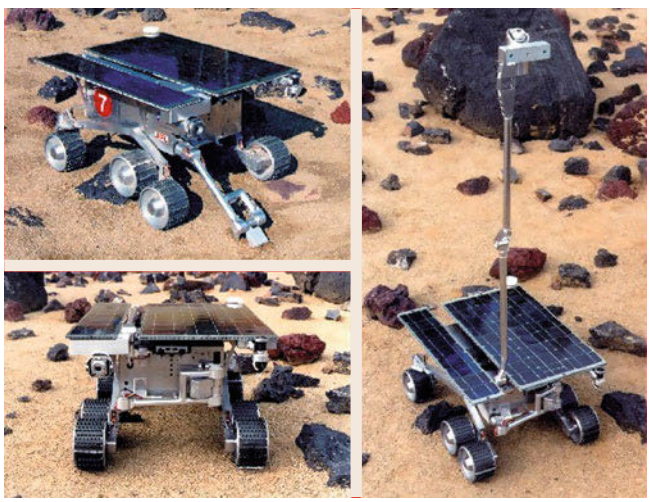


Fig. 55.21 Rocky 7

Automated classification of possible science targets based on clustering of spectral data, figure-ground segmentation of rocks, and other approaches have been attempted with some success. At the time of this writing, some of these technologies have been uploaded to the twin Mars exploration rovers *Spirit* and *Opportunity* (Fig. 55.20) [55.27], including automated detection of temporary events of scientific interest such as dust-devils and clouds [55.44]. The Spirit rover was lost when it broke through a surface crust into a small crater filled with fine dust in late 2009, with attempts to free it continuing until mid 2010. The Mars Science Laboratory *Curiosity* was landed in August 2012 and seeks to explore Mt. Sharpe in the center of Gale Crater.

The ESA with a number of contractors, e.g., ASTRIUM and DLR has been working for several years on a 6-wheeled Mars rover ExoMars that is supposed to be flown after 2018 as well as on a *next lunar lander* NLL Rover, but it is not finally clear, when and in which configuration the mission might be realized.

55.2.4 Sensing and Perception

In the 1980s, most planetary rover sensing research was based on laser ranging or stereo vision. Stereo vision was too computationally intensive for early low-power, radiation-hard processors, so the Sojourner Mars rover used a simple form of laser ranging to determine which areas were safe to traverse. Between the launch of Sojourner (1996) and the Mars exploration rovers (2003) sufficient progress had been made in radiation-hardened flight processors that stereo vision was used for hazard detection on MER [55.45], mostly in experiments conducted with the Rocky-7 rover (Fig. 55.21). This allowed much larger numbers of range points to be incorporated into the hazard-detection algorithm (thousands of points, instead of the 20 discrete range points used by Sojourner). Perception of hazards on Sojourner was based on simple computations of average slope and roughness over the 4×5 array of range measurements, as well as the maximum height differences.

The two MER rovers and the MSL rover use a more sophisticated evaluation of the safety of the rovers along a large number of candidate arcs from its current location. Many other algorithms for perception of terrain hazards have been used with reasonable success by various organizations. Today it is probably fair to say that the *unsolved problems* lie not in the area of geometric hazards (e.g., hazards that can be evaluated completely based on accurate knowledge of the shape of the terrain) but rather in the area of nongeometric hazards (e.g., hazards where uncertainties in

the load-bearing or frictional properties of the terrain determine the safety of a proposed traverse). Accurate estimation of load-bearing or friction properties of terrain by remote sensing is a very challenging task that will not be completely *solved* anytime soon, if ever.

55.2.5 Estimation

Most estimation for planetary surface exploration relates to the internal state of the robot, or its position, pose and kinematic configuration with respect to the environment. Internal state sensors such as encoders on any active or passive articulations in the vehicle are used, along with a kinematic model and inertial sensors such as accelerometers and gyroscopes, to estimate the pose of the vehicle in inertial space. Perceptual algorithms such as surface reconstruction from clouds of range-points as developed by stereo vision put terrain geometry estimates into this same representation. Heading in inertial space is generally the most difficult to reliably estimate, due to the lack of navigation aids such as the global positioning system, or any easily measured heading reference such as a global magnetic field. Integration of rate-gyro data is used to maintain local attitude during motion while accurate estimation of the rotation axis for Mars is possible by long integrations of 3 axis rate-gyro data while the vehicle is stopped. Similar approaches are probably not feasible for the Moon, because of its slow rotation rate. Imaging of the solar disk or constellations of stars at precisely known times can be combined with stored models of the rotation of the planet to allow accurate estimation of the complete pose of the vehicle in inertial space. Kalman filtering or related techniques are generally employed to reduce the effects of measurement noise.

55.2.6 Manipulators for In-Situ Science

The Mars exploration rovers were the first planetary exploration vehicles to have general-purpose manipulators. (Lunokhod had a single-purpose soil penetrometer, and Sojourner had a single degree-of-freedom device to place an Alpha-Proton X-Ray spectrometer in direct contact with the terrain.) The MER arms each have 5°-of-freedom and a reach of over 1 m. Accurate gravity-sag models of the lightweight arm allow the precise position to be predicted in advance of any command to deploy an instrument, and contact sensors allow the arm to stop before any excessive forces build up in the relatively flexible arm. Future arms for planetary surface operations, especially any proposed assembly, maintenance, or servicing tasks as part of the proposed lunar outpost, will require force sensing to protect the stronger but much more rigid arms from damage, as well as to allow controlled forces to be applied to the terrain or workpieces. Of course, there is a huge body of knowledge associated with industrial robot arms, and undersea robotics (e.g., for the offshore oil industry), but such arms are generally very heavy, fast, and stiff compared with credible systems for planetary surface use. Delicate force control has rarely been applied to industrial settings. Space hardware is necessarily very lightweight, and so both the arms and the workpieces will need to have well-resolved force sensing and control to prevent damage to one or both. Because of severe limits on both mass and power, and to avoid unnecessary risk, space manipulation tends to be slow. Historically, this means the gear ratio between each motor and the corresponding output shaft is very large, making the use of motor current as an estimator for output torque very problematic. Other low-mass and robust means for accurate sensing of applied forces in the space environment are needed.

55.3 Mathematical Modeling

Broadly speaking, both on-orbit manipulators and surface mobile robots are considered to be common articulated body systems with a moving base. One point that clearly distinguishes them from other ground-based robots, such as industrial manipulators, is the existence of a moving base.

55.3.1 Space Robot as an Articulated Body System

The robotic systems discussed in this chapter comprise one or multiple articulated limbs mounted on a base body that has a dynamic coupling with these limbs. Typical styles of such moving base systems are categorized into the following groups [55.46].

Free-Floating Manipulator Systems

A space free-flyer that has one or more manipulator arms, as illustrated schematically in Fig. 55.22a, is a typical example of this group. When operating the manipulator arm(s), the position and orientation of the base spacecraft fluctuates due to the manipulator reaction. The kinetic momentum of the system is conserved if no external forces or moments are applied, and the conservation law for this system governs the reaction dynamics. Coordination or isolation, between the base

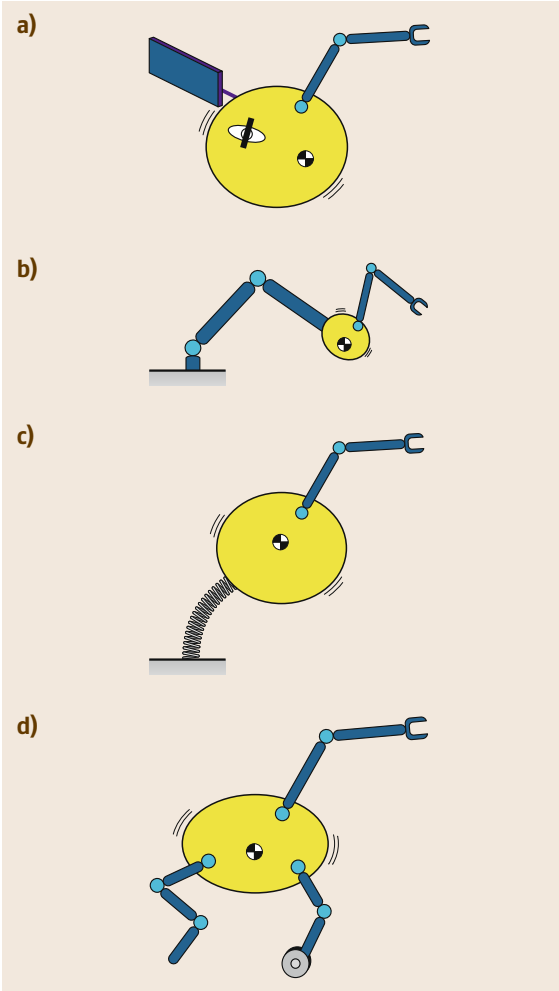


Fig.55.22a–d Four basic types for moving base robots: **(a)** Free-floating manipulator system, **(b)** macro–micro manipulator system, **(c)** flexible structure-based manipulator system, and **(d)** surface locomotive robot system

and manipulator dynamics is key to advanced motion control.

Macro–Micro Manipulator Systems

A robotic system that comprises a relatively small arm (microarm) for fine manipulation mounted on a relatively large arm (macro-arm) for coarse positioning, is called a macro–micro manipulator system. The SSRMS (Canadarm2) and the SPDM (Dextre) system, as well as the JEMRMS on the Japanese module of the ISS, are good examples. Here, the connecting interface at the end point of the macro arm or the root of the micro arm is modeled as the base body (Fig. 55.22b). A free-flying space robot may be treated in this group when its base body is actively controlled by actuators that produce external forces and moments, such as gas-jet thrusters. In this case, these actuators can be modeled as a virtual macro-arm [55.47].

Flexible-Based Manipulator Systems

If the macro arm behaves as a passive flexible (elastic) structure in a macro–micro manipulator system, the system is considered to be a flexible-based manipulator (Fig. 55.22c). Such a situation can be observed in the operation of the ISS, when the SSRMS is servo or brake locked after its coarse operation. Here, the issue is that the base body (the root of the micro arm, or the end of the macro arm, according to the definition above) is subject to vibrations that will be excited by the reaction of the micro arm.

Mobile Robots with Articulated Limbs

Mobile robots for surface locomotion have the same structure in terms of the dynamics equation as the above groups. This group includes wheeled vehicles, walking (articulated limb) robots, and their hybrids. In a wheeled vehicle, suspension mechanisms, if any, are also modeled as articulated limb systems. The forces and moments yielded by contact with ground, or planetary surface, govern the motion of the system (Fig 55.22d).

55.3.2 Equations for Free-Floating Manipulator Systems

Let us first consider a free-floating system with a single or multiple manipulator arm(s) mounted on a base spacecraft. The pioneering work in the mathematical models of this type of space manipulator systems were conducted in late 1980s and early 1990s and collected in the book [55.48], published in 1993. In this section, the models that are widely accepted today are introduced.

The base body, termed link 0, is floating in inertial space without any external forces or moments. At the end point of the arm(s), external forces/moments may apply. For such a system, the equation of motion is expressed as follows

$$\begin{pmatrix} \mathbf{H}_\mathrm{b} & \mathbf{H}_\mathrm{bm} \\ \mathbf{H}_\mathrm{bm}^\mathrm{T} & \mathbf{H}_\mathrm{m} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{x}}_\mathrm{b} \\ \ddot{\boldsymbol{\phi}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_\mathrm{b} \\ \boldsymbol{c}_\mathrm{m} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{pmatrix} + \begin{pmatrix} \mathbf{J}_\mathrm{b}^\mathrm{T} \\ \mathbf{J}_\mathrm{m}^\mathrm{T} \end{pmatrix} \boldsymbol{F}_\mathrm{h} \,. \tag{55.1}$$

The kinematic relationship among $\boldsymbol{x}_\mathrm{h}$, $\boldsymbol{x}_\mathrm{b}$, and $\boldsymbol{\phi}$ is expressed using Jacobian matrices as

$$\dot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,, \tag{55.2}$$

$$\ddot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{J}}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\ddot{\boldsymbol{x}}_\mathrm{b} + \dot{\mathbf{J}}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,. \tag{55.3}$$

Where the notations are listed as follows:

$\boldsymbol{x}_\mathrm{b} \in \mathbb{R}^6$: position/orientation of the base
$\boldsymbol{\phi} \in \mathbb{R}^n$: joint angle of the arm
$\boldsymbol{x}_\mathrm{h} \in \mathbb{R}^{6k}$: position/orientation of the end point(s)
$\boldsymbol{\tau} \in \mathbb{R}^n$: joint torque of the arm
$\boldsymbol{F}_\mathrm{h} \in \mathbb{R}^{6k}$: external forces/moments on the end point(s)
n: number of total joints
k: number of manipulator arms,

and $\mathbf{H}_\mathrm{b}$, $\mathbf{H}_\mathrm{m}$, and $\mathbf{H}_\mathrm{bm}$ are inertia matrices for the base body, manipulator arm, and the coupling between the base and the arm, respectively, and $\boldsymbol{c}_\mathrm{p}$ and $\boldsymbol{c}_q$ are nonlinear Coriolis and centrifugal forces, respectively.

For a free-floating manipulator in orbit, the gravity forces exerted on the system can be neglected, and so the nonlinear term becomes

$$\boldsymbol{c}_\mathrm{b} = \dot{\mathbf{H}}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} + \dot{\mathbf{H}}_\mathrm{bm}\dot{\boldsymbol{\phi}}_\mathrm{b} \,.$$

Integrating the upper set of equation in (55.1) with respect to time, we obtain the total momentum of the system as

$$\mathcal{L} = \int \mathbf{J}_\mathrm{b}^\mathrm{T}\boldsymbol{F}_\mathrm{h}\,\mathrm{d}t = \mathbf{H}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} + \mathbf{H}_\mathrm{bm}\dot{\boldsymbol{\phi}} \,. \tag{55.4}$$

For the case in which reaction wheels are mounted on the base body, they are included as additional manipulator arms.

55.3.3 Generalized Jacobian and Inertia Matrices

From (55.2) and (55.4), the coordinates of the manipulator base $\dot{\boldsymbol{x}}_\mathrm{b}$, which are passive and unactuated coordinates, can be eliminated, as follows

$$\dot{\boldsymbol{x}}_\mathrm{h} = \hat{\mathbf{J}}\dot{\boldsymbol{\phi}} + \dot{\boldsymbol{x}}_\mathrm{h0} \,, \tag{55.5}$$

where

$$\hat{\mathbf{J}} = \mathbf{J}_\mathrm{m} - \mathbf{J}_\mathrm{b}\mathbf{H}_\mathrm{b}^{-1}\mathbf{H}_\mathrm{bm} \,, \tag{55.6}$$

and

$$\dot{\boldsymbol{x}}_\mathrm{h0} = \mathbf{J}_\mathrm{b}\mathbf{H}_\mathrm{b}^{-1}\mathcal{L} \,. \tag{55.7}$$

Since $\mathbf{H}_\mathrm{b}$ is the inertia tensor of a single rigid body (the manipulator base), it is always positive definite, and so its inverse exists.

The matrix $\hat{\mathbf{J}}$ was first introduced in [55.49, 50] and is referred to as the *generalized jacobian matrix*. In its original definition, it was assumed that no external forces/moments acted on the system. If $\boldsymbol{F}_\mathrm{h} = \mathbf{0}$, then the term $\dot{\boldsymbol{x}}_\mathrm{h0}$ becomes constant, and, in particular, if the system has zero initial momentum $\dot{\boldsymbol{x}}_\mathrm{h0} = \mathbf{0}$, then (55.5) becomes simple. However, note that in the derivation of (55.6), zero or constant momentum is not a necessary condition.

Using this matrix, the manipulator hand can be operated under a resolved motion-rate control or resolved acceleration control in inertial space. Thanks to the generalized Jacobian, although the reactive base motion occurs during the operation, the hand is not disturbed by the motion.

From the upper and lower sets of equations in (55.1), $\ddot{\boldsymbol{x}}_\mathrm{b}$ can be eliminated to obtain the following expression

$$\hat{\mathbf{H}}\ddot{\boldsymbol{\phi}} + \hat{\boldsymbol{c}} = \boldsymbol{\tau} + \hat{\mathbf{J}}\boldsymbol{F}_\mathrm{h} \,, \tag{55.8}$$

where

$$\hat{\mathbf{H}} = \mathbf{H}_\mathrm{m} - \mathbf{H}_\mathrm{bm}^\mathrm{T}\mathbf{H}_\mathrm{b}^{-1}\mathbf{H}_\mathrm{bm} \,. \tag{55.9}$$

The matrix $\hat{\mathbf{H}}$ is known as the *generalized inertia matrix* for space manipulators [55.48]. This matrix represents the inertia property of the system in the joint space and can be mapped onto the Cartesian space using the generalized Jacobian matrix

$$\hat{\mathbf{G}} = \hat{\mathbf{J}}\hat{\mathbf{H}}^{-1}\hat{\mathbf{J}}^\mathrm{T} \,. \tag{55.10}$$

The matrix $\hat{\mathbf{G}}$ is referred to as the *inversed inertia tensor* for space manipulators and is useful for the discussion of impact dynamics when a space manipulator collides with or captures a floating target in orbit [55.51].

The generalized Jacobian matrix (GJM) is a useful concept, with which the manipulator end point can perform positioning or trajectory tracking control by a simple control algorithm regardless of the attitude deviation during the operation.

A simplified laboratory demonstration was carried out using a two-dimensional free-floating test bed called *EFFORTS* [55.52]. To simulate the motion in a microgravity environment, a robot model was floated on a thin film of pressurized air on a horizontal plate, so that frictionless motion with momentum conservation was achieved.

Figure 55.23 depicts the test bed and a typical experimental result. For Fig. 55.23b, the control command was given to the floating robot by

$$\dot{\boldsymbol{\phi}} = \hat{\mathbf{J}}^{-1}\dot{\boldsymbol{x}}_\mathrm{d} \,, \tag{55.11}$$

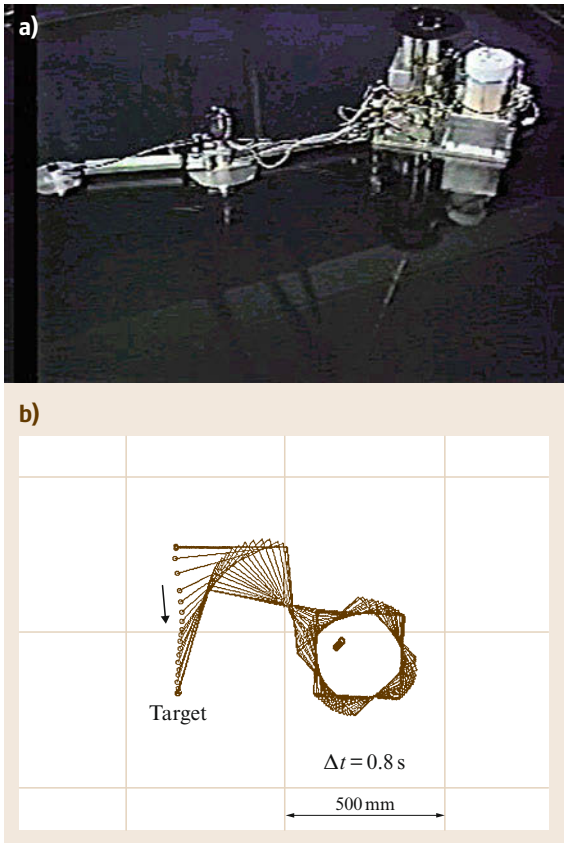


Fig.55.23a,b Laboratory test bed for a free-floating space robot: **(a)** The EFFORTS test bed, **(b)** a target capture result

where $\dot{\boldsymbol{x}}_{\mathrm{d}}$ is the desired velocity of the manipulator end point, the value of which was given and updated by on-line measurement of the end point position $\boldsymbol{x}_{\mathrm{h}}$ and the target position $\boldsymbol{x}_{\mathrm{t}}$, as follows

$$\dot{\boldsymbol{x}}_{\mathrm{d}} = \frac{\boldsymbol{x}_{\mathrm{t}} - \boldsymbol{x}_{\mathrm{h}}}{\Delta t} , \tag{55.12}$$

where Δt is the time interval of the on-line control loop. The desired end point velocity was simply resolved into the joint velocity by (55.11).

The result clearly shows that the manipulator end point properly reached the target in an optimum manner, although the robot base rotated considerably due to the manipulator reaction. Note that, since the target was stationary in this example, the resulting motion trace was a straight line. However, thanks to the on-line control, the manipulator was also able to track and reach a moving target with the same control law.

The validity and effectiveness of the GJM-based manipulator control were also demonstrated in orbit by Japanese ETS-VII mission [55.14].

55.3.4 Linear and Angular Momenta

The integral of the upper set of equation in (55.1) gives a momentum equation, as shown in (55.4), which is composed of the linear and angular momenta. The linear part is expressed as

$$\breve{\mathbf{H}}_{\mathrm{b}} \boldsymbol{v}_{\mathrm{b}} + \breve{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}} = \boldsymbol{P} , \tag{55.13}$$

where $\boldsymbol{v}_{\mathrm{b}}$ is the linear velocity of the base, $\boldsymbol{P}$ is the initial linear momentum, and the inertia matrices with the mark of $\breve{(\cdot)}$ are the corresponding components for the linear momentum [55.48]. When the linear momentum is further integrated, the result verifies the principle that the mass centroid of the entire system either remains stationary or translates with a constant velocity.

The angular momentum equation, however, does not have a second integral, and therefore provides a first-order nonholonomic constraint [55.53]. The equation can be expressed as

$$\tilde{\mathbf{H}}_{\mathrm{b}} \boldsymbol{\omega}_{\mathrm{b}} + \tilde{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}} = \boldsymbol{L} , \tag{55.14}$$

where $\boldsymbol{\omega}_{\mathrm{b}}$ is the angular velocity of the base, $\boldsymbol{L}$ is the initial angular momentum, and the inertia matrices with the mark of $\tilde{(\cdot)}$ are the corresponding components for the angular momentum [55.48]; $\tilde{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}}$ represents the angular momentum generated by the manipulator motion.

Equation (55.14) can be solved for $\boldsymbol{\omega}_{\mathrm{b}}$ with zero initial angular momentum

$$\boldsymbol{\omega}_{\mathrm{b}} = -\tilde{\mathbf{H}}_{\mathrm{b}}^{-1} \tilde{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}} . \tag{55.15}$$

This expression describes the resulting disturbance motion of the base when there is joint motion $\dot{\boldsymbol{\phi}}$ in the manipulator arm.

There are a number of points worth discussion when analyzing this equation. The magnitudes and directions of the maximum and minimum disturbances can be obtained from the singular value decomposition of the matrix $\left(-\tilde{\mathbf{H}}_{\mathrm{b}}^{-1} \tilde{\mathbf{H}}_{\mathrm{bm}}\right)$ and displayed on the map. Such a map is called a *disturbance map* [55.54, 55]. Equation (55.15) is also used for the feed-forward compensation in the coordinated manipulator/base control model [55.56, 57].

55.3.5 Virtual Manipulator

The concept of the *virtual manipulator* (VM) is an augmented kinematic representation that considers the base motion due to reaction forces or moments. The model is based on the fact that the mass centroid of the entire system does not move in the free-floating system without any external forces [55.58]. The mobility

of the end point of the arm is decreased by the base motion. In the VM representation, such mobility degradation is expressed by virtually shrinking the length of the real arm according to the mass property. Note that the VM considers only linear momentum conservation. If the differential expression of VM is obtained using a Jacobian matrix, the Jacobian is not a conventional kinematic Jacobian, but rather a version of the generalized Jacobian defined by the combination of the kinematic equation (55.2), and the linear momentum equation (55.13).

55.3.6 Dynamic Singularity

Dynamic singularities are singular configurations in which the manipulator end point loses mobility in some inertial direction [55.59]. Dynamic singularities are not found in earth-based manipulators, but rather occur in free-floating space manipulator systems due to the coupling dynamics between the arm and the base. Dynamic singularities coincide with the singularities of the generalized Jacobian matrix determined by (55.6). The singular value decomposition (SVD) of a manipulator Jacobian matrix provides *manipulability* analysis. Likewise, the SVD of the generalized Jacobian matrix yields the manipulability analysis of a free-floating space manipulator [55.60]. Figure 55.24 shows the comparison of the manipulability distribution between a 2-DOF ground-based manipulator and a 2-DOF floating manipulator, from which the degradation of the manipulability is observed in the space arm due to the dynamic coupling.

55.3.7 Reaction Null-Space (RNS)

From a practical point of view, any change in the base attitude is undesirable. As such, manipulator motion planning methods that minimize the base attitude disturbance have been investigated extensively. Analysis of the angular momentum equation reveals that the ultimate goal of achieving zero disturbance is possible.

The following is the angular momentum equation with zero initial angular momentum $\boldsymbol{L} = \boldsymbol{0}$ and the zero attitude disturbance $\boldsymbol{\omega}_{\mathrm{b}} = \boldsymbol{0}$ given in (55.14)

$$\tilde{\mathbf{H}}_{\mathrm{bm}}\dot{\boldsymbol{\phi}} = \boldsymbol{0}\,. \tag{55.16}$$

This equation yields the following null-space solution

$$\dot{\boldsymbol{\phi}} = (\mathbf{I} - \tilde{\mathbf{H}}_{\mathrm{bm}}^{+}\tilde{\mathbf{H}}_{\mathrm{bm}})\dot{\boldsymbol{\zeta}}\,. \tag{55.17}$$

The joint motion given by (55.17) is guaranteed not to disturb the base attitude. Here, the vector $\dot{\boldsymbol{\zeta}} \in \mathbb{R}^n$ is arbitrary and the null-space of the inertia matrix $\tilde{\mathbf{H}}_{\mathrm{bm}} \in \mathbb{R}^{3\times n}$ is called the *reaction null-space (RNS)* [55.61].

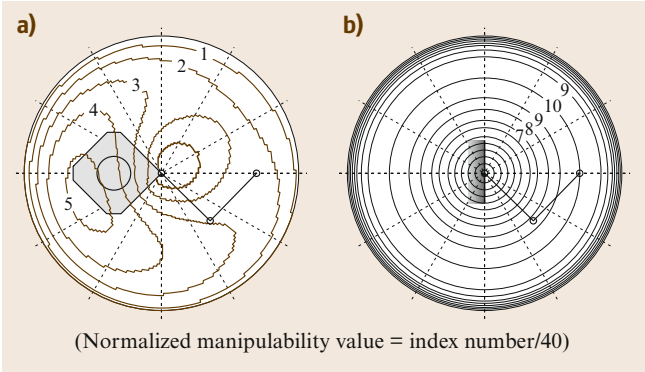


Fig.55.24a,b Manipulability distribution in the work space (after [55.60])

The DOF for $\dot{\boldsymbol{\zeta}}$ is $n-3$. For example, if a manipulator arm mounted on a free-floating space robot has 6-DOF, i. e., $n = 6$, then 3-DOF remain in the reaction null-space. These DOFs can be specified by introducing additional motion criteria, such as end point positioning of the arm. Such manipulator operation that produces no reaction in the base is called the *reactionless manipulation* [55.62].

The validity and effectiveness of the RNS-based reactionless manipulation were demonstrated in orbit by Japanese ETS-VII mission [55.14].

55.3.8 Motion Planning Issues

This subsection addresses the generation of feasible trajectories for a free-flying robot for executing typical point-to-point or grasping tasks. The subject falls under the problem domain of motion planning, with the aim of satisfying motion constraints which generally cannot be satisfied with use of local methods alone (feedback control and model predictive control). A trajectory resulting from the motion planning is fed to a tracking controller, which accounts for any modeling errors, to accomplish the task in question. This approach also aims at providing autonomous skills for supporting a human ground operator.

A typical task of interest here is that of a point-to-point maneuver of a robot manipulator mounted on a servicer satellite, to bring its end-effector in some desired inertial position and orientation. This task may require actuation of the servicer, or may, if possible, be preferably executed in free-floating mode, to avoid issues related to the use of the on-board thrusters, such as fuel expenditure. In this context, to minimize the attitude change of the servicer resulting from the manipulator motion, a noticeable fundamental result for a point-to-point maneuver, derived from nonlinear optimization theory, is the V-maneuver [55.63]. Intuitively,

the attitude change is minimized by making the robot first move radially inward, toward the system center of mass, before turning and then radially outward, to reach the new desired final position.

A second task that has received much attention is the grasping of a free tumbling target, like a defected satellite. Currently, there are a number of space programs worldwide which are addressing this task, like the preparation of the demonstration mission DEOS of the DLR (Sect. 55.1.2) for grasping a small satellite in LEO, the e.Deorbit study of the ESA for the deorbiting of the defected ENVISAT satellite and PHOENIX of DARPA for grasping a geostationary satellite in the graveyard orbit. Different approaches can be found in the literature to tackle this problem, to include [55.64, 65] for feedback control, [55.66] for model predictive control, [55.67] for optimal control in Cartesian space and [55.68] for nonlinear optimization, to mention some.

Due to the nonholonomic nature of the dynamics of a free-floating robot (Sect. 55.3.4), in order to satisfy all relevant motion constraints, the motion planning problems above can only be solved through numerical integration. Note in fact, that the final system configuration for a given final end-effector position in inertial space is function of the whole path taken by the robot throughout the motion.

Principally, the motion planning tasks above may be formulated as an optimal control problem of the type

$$\min_{t_\mathrm{f},\boldsymbol{\phi}(t)} \Gamma(\boldsymbol{\phi}(t),\boldsymbol{\tau}(t),t_\mathrm{f})\,, \tag{55.18}$$

subject to

$$g(t_\mathrm{f},\boldsymbol{\phi}(t)) = 0 \tag{55.19}$$

$$h(t_\mathrm{f},\boldsymbol{\phi}(t)) \le 0\,, \tag{55.20}$$

for $0 \le t \le t_\mathrm{f}$ and where t_f is the final time; $\boldsymbol{\phi}(t)$ is the vector of joint positions; $\boldsymbol{\tau}(t)$ is the vector of joint torques; Γ is a predefined cost function; g are equality constraints to include, for example, the state transition equations; h are inequality constraints, for example the joint box constraints on position, velocity and torque, or collision avoidance constraints. Other motion constraints may include inequality constraints on the end-effector forces during contact, or other operational constraints.

This consists of an infinite dimensional problem, in the given time interval, which generally cannot be solved in closed form. The direct shooting optimization methods lend themselves well to solving these problems iteratively [55.69], where for example the independent DOF (in the case of a free-floating system, the robot joint angles) are parameterized in time as

$$\boldsymbol{\phi} = \boldsymbol{\phi}(t,\boldsymbol{p}) \tag{55.21}$$

with $\boldsymbol{p} \in \mathbb{R}^n$, for n optimization parameters, where $\boldsymbol{\phi}$ may be a polynomial function, a B-spline function, or any other. This way, the problem becomes that of finding a suitable value for the parameters $\boldsymbol{p}$ which satisfies all motion constraints (feasibility), as well as perhaps minimizes a cost function (optimality), like the time-to-collision [55.70], the end time [55.67], or the mechanical energy [55.68]. The problem is as such transformed into a finite dimensional problem and can be solved as a nonlinear programming problem (NLP) with classical numerical iterative methods such as sequential quadratic programming (SQP).

For example, for a Cartesian point-to-point problem, we have the following supplementary equality constrains at the end time

$$\boldsymbol{X}^\mathrm{e} = \int_0^{t_\mathrm{f}} \hat{\mathbf{J}}\dot{\boldsymbol{\phi}}(t,\boldsymbol{p})\,\mathrm{d}t = \boldsymbol{X}^\mathrm{e}_\mathrm{des}\,, \tag{55.22}$$

where $\boldsymbol{X}^\mathrm{e} \in \mathbb{R}^n$ denotes the end-effector pose (array matrix of dimension DOFe $\times$ 1). Note that the constraint itself has $\infty^{\mathrm{DOF+3-DOFe}}$ solutions for the system configuration (for DOF robot joints), however these all imply a specific base body attitude as well as a specific robot configuration. Solving the problem of reaching a specific system configuration is a hard nonholonomic control problem, which is avoided by solving the constraint thorough the integral above and an adequate parameter set $\boldsymbol{p}$.

Extensions of this are necessary to treat the grasping task. The latter can ideally be separated into three phases: approach, tracking and, stabilization. The first comprises a point-to-point maneuver, however with a nonzero end velocity. The second comprises a tracking maneuver, in which the Cartesian motion of the robot end-effector is dictated by the tumbling motion and the geometry of the target as it follows the grasping point and homes in onto it, to finally grasp it. Note that this phase aims at minimizing the impact between the end-effector and the target. The third phase involves a robot joint velocity decay maneuver, once the target is grasped. This formulation results in a multiple-phase problem, for which the boundaries between the phases introduce supplementary motion constraints in the motion planning problem.

From a methodological point of view, note that there is no simple measure to determine if and when the grasping point on the target will be reachable from the current robot configuration (Fig. 55.25) and whether

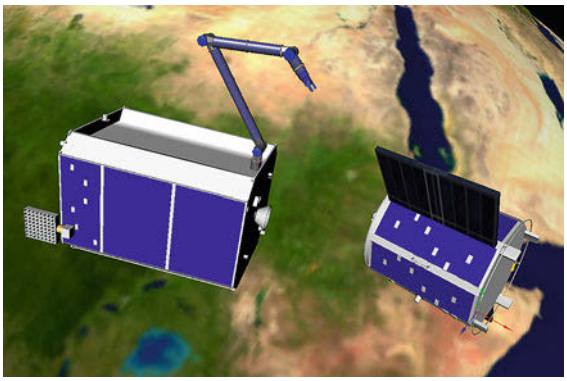


Fig. 55.25 Orbital scenario: servicer satellite with 7-DOF manipulator and target satellite with solar panels. Coordinate system of a predefined grasping point on target ring shown

the trajectory which derives from a local control law will be feasible at all times (accounting for the motion constraints listed above). It is also necessary to provide information on the time synchronization between the motion of the grasping point on the target and that of the robot. Furthermore, the nonlinear nature of the robot kinematics needs to be exploited to favor a successful grasp. These considerations speak in favor of the use of a reference trajectory, which is computed off-line by means of a global search, based on a motion prediction of the target and on a geometric model of the same target and of the space robot [55.68].

It is also well known that optimization methods suffer from convergence issues (arising from local minima), if a judicious initial guess is not available. It is for this reason that a look-up table approach is necessary [55.68] in order to provide a sufficiently high probability of convergence for a given grasping task.

The necessity of a look-up table also arises from the long computation times of the optimization process, which result from the aforementioned necessity to integrate the equations of motion for any given optimization iteration. This time is generally reduced if a solution close to the sought one is given as a starting point. An attempt to eliminate the necessity of integrating the equations of motion was also made in [55.71], where a differentially flat representation for a free-floating robot was sought. In such a formulation, there are as many differential equations as there are independent state variables, or flat variables, and as such, any parameterization of the flat variables is solution of the equations of motion. Such a representation was found for the case in which the robot has three joints and the center of mass of its load lies on the rotation axis of the last joint.

It is also of interest to make some considerations on practical technological issues which can influence one or the other future research direction. With regard to the minimization of the servicer attitude change during robotic operations, it is worth realizing that this is generally only an issue for maintaining the communication link from low Earth orbit with a geostationary satellite, for which a high pointing accuracy is required (note that for a communication link from low Earth orbit to ground instead, an omnidirectional antenna is sufficient). Therefore, rather than limiting the robot workspace to its reaction null-space to avoid any attitude change at all, in which case the resulting robot movements would generally be confined to the extent of becoming of little practical use, a simple technological solution is possible through the implementation of a gimbal joint for the communication antenna on the servicer.

Another simple technological solution to a theoretically involved problem is that of controlling the attitude of the servicer by means of adequate closed-loop maneuvers of the robot. As is well known, reaction wheels on a satellite achieve the same result, also with the same principle of momentum transfer, but however with a far simpler control law than that necessary for a robot. It is then important to realize that, although reaction wheels were until today far too small for any useful robotic application, larger reaction wheels are currently being developed, for example, in the context of the BIROS mission of the DLR, which is planned to fly in 2014. Although these will still not be able to fully compensate the typical robot dynamic coupling terms (if wanting to fully stabilize the servicer attitude), they will however be able to allow significant attitude slew maneuvers of the servicer quickly, without having to resort to the use of the robot or of the thrusters. Furthermore, they will be useful to reduce the robot singularities, due to the enhanced system's actuation.

55.3.9 Equations for Flexible-Based Manipulator Systems

Next, let us consider a flexible-based manipulator system in which a single or multiple manipulator arm(s) are supported by a flexible beam or a spring and damper (visco-elastic) system. For such a system, the equation of motion is obtained using the following variables:

$\boldsymbol{x}_\mathrm{b} \in \mathbb{R}^6$: position/orientation of the base
$\boldsymbol{\phi} \in \mathbb{R}^n$: joint angle of the arm
$\boldsymbol{x}_\mathrm{h} \in \mathbb{R}^{6k}$: position/orientation of the end point(s)
$\boldsymbol{F}_\mathrm{b} \in \mathbb{R}^6$: forces/moments to deflect the flexible base
$\boldsymbol{\tau} \in \mathbb{R}^n$: joint torque of the arm
$\boldsymbol{F}_\mathrm{h} \in \mathbb{R}^{6k}$: external force/torque on the end point(s).

$$\begin{pmatrix} \mathbf{H}_\mathrm{b} & \mathbf{H}_\mathrm{bm} \\ \mathbf{H}_\mathrm{bm}^\mathrm{T} & \mathbf{H}_\mathrm{m} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{x}}_\mathrm{b} \\ \ddot{\boldsymbol{\phi}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_\mathrm{b} \\ \boldsymbol{c}_\mathrm{m} \end{pmatrix} = \begin{pmatrix} \boldsymbol{F}_\mathrm{b} \\ \boldsymbol{\tau} \end{pmatrix} + \begin{pmatrix} \mathbf{J}_\mathrm{b}^\mathrm{T} \\ \mathbf{J}_\mathrm{m}^\mathrm{T} \end{pmatrix} \boldsymbol{F}_\mathrm{h} \,, \tag{55.23}$$

$$\dot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,, \tag{55.24}$$

$$\ddot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{J}}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\ddot{\boldsymbol{x}}_\mathrm{b} + \dot{\mathbf{J}}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,. \tag{55.25}$$

Here, with the gravitational force $\boldsymbol{g}$ in Cartesian space, the term $\boldsymbol{c}_\mathrm{b}$ is generally expressed as

$$\boldsymbol{c}_\mathrm{b} = \boldsymbol{f}(\mathbf{x}_\mathrm{b}, \boldsymbol{\phi}, \dot{\boldsymbol{x}}_\mathrm{b}, \dot{\boldsymbol{\phi}}) + \boldsymbol{g}(\boldsymbol{x}_\mathrm{b}, \boldsymbol{\phi}) \,. \tag{55.26}$$

The difference from the equation of a free-floating manipulator system (55.1) is the existence of the base constraint force $\boldsymbol{F}_\mathrm{b}$. Let $\mathbf{D}_\mathrm{b}$ and $\mathbf{K}_\mathrm{b}$ be the matrix representing the damping and spring factors, respectively, of the flexible-base. The constraint forces and moments $\boldsymbol{F}_\mathrm{b}$ are then expressed as

$$\boldsymbol{F}_\mathrm{b} = -\mathbf{D}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} - \mathbf{K}_\mathrm{b}\Delta\boldsymbol{x}_\mathrm{b} \,. \tag{55.27}$$

Since the base is constrained, the total momentum is not conserved, and it might be meaningless to check the system momentum. However, it is important to consider the partial momentum $\mathcal{L}_\mathrm{m}$ for the part of the manipulator arm

$$\mathcal{L}_\mathrm{m} = \mathbf{H}_\mathrm{bm}\dot{\boldsymbol{\phi}} \,, \tag{55.28}$$

which is termed the *coupling momentum* [55.72]. Its time derivative describes the forces and moments $\boldsymbol{F}_\mathrm{m}$, which are yielded by the dynamic reaction from the manipulator arm to the base

$$\boldsymbol{F}_\mathrm{m} = \mathbf{H}_\mathrm{bm}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{H}}_\mathrm{bm}\dot{\boldsymbol{\phi}} \,. \tag{55.29}$$

Using $\boldsymbol{F}_\mathrm{b}$ and $\boldsymbol{F}_\mathrm{m}$, the upper set of equation in (55.23) is rearranged as

$$\mathbf{H}_\mathrm{b}\ddot{\boldsymbol{x}}_\mathrm{b} + \mathbf{D}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} + \mathbf{K}_\mathrm{b}\Delta\boldsymbol{x}_\mathrm{b} = -\boldsymbol{g} - \boldsymbol{F}_\mathrm{m} + \mathbf{J}_\mathrm{b}^\mathrm{T}\boldsymbol{F}_\mathrm{h} \,. \tag{55.30}$$

Equation (55.23) or (55.30) is a familiar expression for flexible-base manipulators [55.73, 74].

55.3.10 Advanced Control for Flexible Structure Based Manipulators

In this subsection, advanced control issues for a macro–micro space manipulator are discussed. The SSRMS attached to the SPDM (Fig. 55.4) and the JEMRMS (Fig. 55.5) are examples. Operation modes for this class of space manipulators include coarse motion by the macro (long-reach) component and fine manipulation by the micro component. In normal cases, these two control modes are executed exclusively. Namely, while one component is active the other component should be servo (or brake) locked. Thus, during the operation of the micro component, the macro component behaves just as a passive base.

Due to the flexible nature of the space long-reach arm, the macro part is subject to vibrations. These vibrations can be excited during coarse positioning and may remain for a long time after each operation. In fine manipulation, the macro arm behaves as a passive flexible structure, but then vibrations can be excited by reactions from the motion of the micro arm. These motions degrade the control accuracy and operational performance of the system. In practice, the booms are usually sufficiently stiff, but flexibility comes mainly from the low stiffness at the joints and gear trains. Moreover, lightweight and microgravity characteristics make the structure sensitive to yield vibrations, and the surrounding vacuum, or the lack of air viscosity, provides a reduced damping effect to the structure.

Conventionally, the vibration issue has been managed for SRMS and SSRMS by the operational skill of well-trained astronauts and by limiting the maximum operational velocity according to the inertia of the handling object. However, if an advanced controller is introduced on the ISS, the training time for astronauts will be reduced and the operational speed can be increased.

Here, the following two subtasks are considered in dealing with this issue:

- Suppression of the vibrations of the flexible base
- End point control in the presence of vibrations.

To suppress the vibrations of the macro arm (flexible base), the coupled dynamics is effectively used. Such control is called *coupled vibration suppression control* [55.75]. The coupled dynamics is a solution space of the micro arm motion with maximum coupling with the vibration dynamics of the macro arm. Note that this solution space is perpendicular to the reaction null-space introduced in Sect. 55.3.7. Since the spaces are orthogonal, the coupled vibration suppression control and reactionless manipulation can be superimposed without any mutual interference.

The motion command of the micro arm to suppress the vibrations is determined with a feedback of the linear and angular velocity of the end point of the macro arm $\dot{\boldsymbol{x}}_\mathrm{b}$

$$\ddot{\boldsymbol{\phi}} = \mathbf{H}_\mathrm{bm}^{+}\mathbf{H}_\mathrm{b}\mathbf{G}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,, \tag{55.31}$$

where $(\cdot)^+$ denotes the right pseudo-inverse, and $\mathbf{G}_\mathrm{b}$ is a positive definite gain matrix.

If written in the form of a joint torque input, the vibration control law is expressed as follows

$$\boldsymbol{\tau} = \mathbf{H}_{\mathrm{m}}\mathbf{H}_{\mathrm{bm}}^{+}\mathbf{G}_{\mathrm{b}}\dot{\boldsymbol{x}}_{\mathrm{b}} \,. \tag{55.32}$$

In the presence of redundancy in the micro arm, (55.31) can be extended to control with a null-space component

$$\ddot{\boldsymbol{\phi}}_{\mathrm{m}} = \mathbf{H}_{\mathrm{bm}}^{+}(\mathbf{H}_{\mathrm{b}}\mathbf{G}_{\mathrm{b}}\dot{\boldsymbol{x}}_{\mathrm{b}} - \dot{\mathbf{H}}_{\mathrm{bm}}\dot{\boldsymbol{\phi}}_{\mathrm{m}}) + \boldsymbol{P}_{\mathrm{RNS}}\boldsymbol{\zeta} \,, \tag{55.33}$$

where $\boldsymbol{\zeta}$ is an n-DOF arbitrary vector and $\boldsymbol{P}_{\mathrm{RNS}} = (\mathbf{E} - \mathbf{H}_{\mathrm{bm}}^{+}\mathbf{H}_{\mathrm{bm}})$ is a projector onto a null space of the coupling inertia matrix $\mathbf{H}_{\mathrm{bm}}$. When the micro arm is operated using (55.33), the closed-loop system is expressed as

$$\mathbf{H}_{\mathrm{b}}\ddot{\boldsymbol{x}}_{\mathrm{b}} + \mathbf{H}_{\mathrm{b}}\mathbf{G}_{\mathrm{b}}\dot{\boldsymbol{x}}_{\mathrm{b}} + \mathbf{K}_{\mathrm{b}}\Delta\boldsymbol{x}_{\mathrm{b}} = \boldsymbol{F}_{\mathrm{b}} + \mathbf{J}_{\mathrm{b}}^{\mathrm{T}}\boldsymbol{F}_{\mathrm{h}} \,. \tag{55.34}$$

Equation (55.34) represents a second-order damped vibration system. With no force input, i. e., $\boldsymbol{F}_{\mathrm{b}} = \boldsymbol{F}_{\mathrm{h}} = 0$, the vibrations converge to zero with a proper choice of the gain matrix $\mathbf{G}_{\mathrm{b}}$.

For the determination of vector $\boldsymbol{\zeta}$, feedback control to reduce the positioning error of the micro arm end point is considered. The error vector is defined as

$$\tilde{\boldsymbol{x}}_{\mathrm{h}} = \boldsymbol{x}_{\mathrm{h}}^{d} - \boldsymbol{x}_{\mathrm{h}} \,. \tag{55.35}$$

After some derivation, the control law for the joint torque of the micro arm is obtained in the following form [55.75, 76]

$$\boldsymbol{\tau} = ((\mathbf{J}_{\mathrm{m}}^{\mathrm{T}})^{+}\mathbf{H}_{\mathrm{m}}\boldsymbol{P}_{\mathrm{RNS}})^{+}(\mathbf{K}_{\mathrm{p}}\tilde{\boldsymbol{x}}_{\mathrm{h}} - \mathbf{K}_{\mathrm{d}}\mathbf{J}_{\mathrm{m}}\dot{\boldsymbol{\phi}}) - \mathbf{G}_{\mathrm{m}}\dot{\boldsymbol{\phi}} \,, \tag{55.36}$$

where $\mathbf{K}_{\mathrm{p}}$, $\mathbf{K}_{\mathrm{d}}$, and $\mathbf{G}_{\mathrm{m}}$ are positive definite gain matrices.

Figure 55.26 shows a block diagram for the control system described by (55.32) and (55.36).

As a simplified demonstration, a planar system with a four-joint redundant manipulator arm atop a flexible beam is considered. Figure 55.27 shows the vibration amplitude of the flexible beam after an impulsive external force. The graph labeled *w/o vs* depicts the vibrations of the beam without any manipulator control but with natural damping. The graph labeled *with vs* depicts the case in which vibration suppression control given by (55.32) is applied, where the vibrations are damped quickly.

In addition, Fig. 55.28 shows the end point motion of the manipulator arm during the control. The graph labeled *w/o RNS* is the case of using (55.32), where the base vibrations were successfully suppressed but

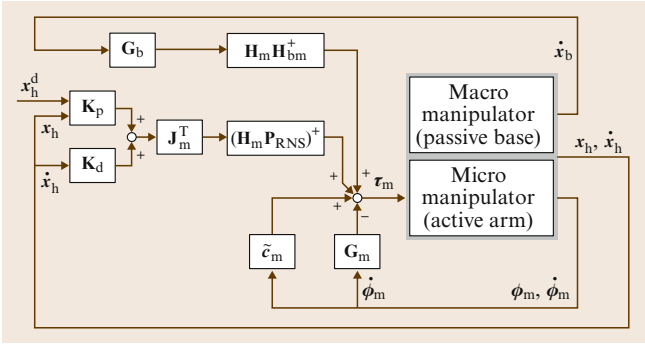


Fig. 55.26 Block diagram for simultaneous vibration suppression and manipulator end point control for a flexible-structure-mounted manipulator system

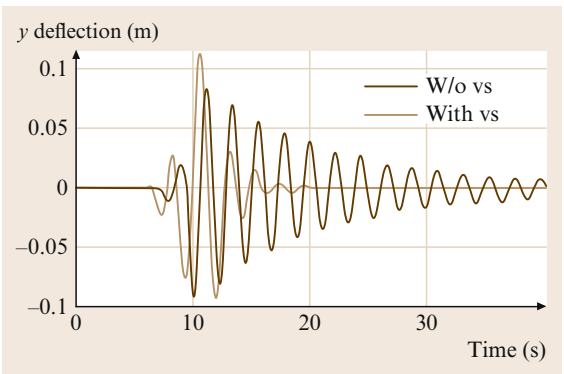


Fig. 55.27 Vibrations of the flexible base

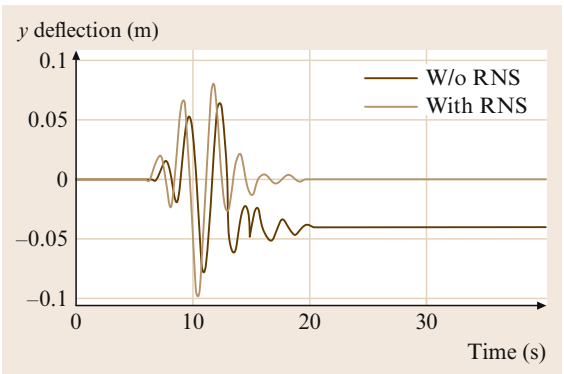


Fig. 55.28 Positioning error of the manipulator end tip

the position of the manipulator end point was deflected by this suppression behavior. The graph labeled *with RNS* depicts the case in which both vibration suppression control given by (55.32) and the end point control given by (55.36) are applied simultaneously. This last case shows that the vibrations were damped successfully and that the positioning error of the manipulator

end point converged to zero. This is a result of the redundancy of the arm.

Here, note that the proposed control method requires precise information on dynamic characteristics, such as the inertia parameters of the arms and the handled payload, if any. To achieve more practical applications, the proposed method must be extended to schemes for parameter identification [55.77] and adaptive control [55.78], with which the convergence of the control is guaranteed even with imprecise a priori knowledge of the dynamic parameters [55.76].

55.3.11 Contact Dynamics and Impedance Control

The capture and retrieval operation of a floating and tumbling *target*, such as a malfunctioning satellite, by a manipulator arm mounted on a servicing robot (called a *chaser*) can be decomposed into the following three phases:

1. Approaching phase (before contact with the target)
2. Contact/impact phase (at the moment of contact)
3. Post contact phase (after contact or grasping).

If the contact is impulsive, the first and the third phases are discontinued by the impulsive phenomena of the second phase. Understanding of this impulsive phenomena is indispensable when designing a comprehensive capture control scheme. In this subsection, the formulation of impact dynamics is first considered, and impedance control (which is useful to minimize the impact forces and prolong the contact duration) is then discussed.

Let us consider a chain of rigid links composed of $n+1$ bodies freely floating in inertial space. As discussed in Sect. 55.3.3, the equation of motion for this type of system becomes (55.8). Here, the impulsive contact force is assumed to be applied at the manipulator end tip and is expressed as $\boldsymbol{F}_\mathrm{h} = (\boldsymbol{f}_\mathrm{h}^\mathrm{T}, \boldsymbol{n}_\mathrm{h}^\mathrm{T})^\mathrm{T} \in \mathbb{R}^6$. This impulsive force also yields a change in the system momenta $(\boldsymbol{P}_\mathrm{g}^\mathrm{T}, \boldsymbol{L}_\mathrm{g}^\mathrm{T})^\mathrm{T} \in \mathbb{R}^6$, expressed as

$$\begin{aligned}\begin{pmatrix}\dot{\boldsymbol{P}}_\mathrm{g}\\ \dot{\boldsymbol{L}}_\mathrm{g}\end{pmatrix} &= \begin{pmatrix} w\mathbf{E} & 0\\ 0 & \mathbf{I}_\mathrm{g}\end{pmatrix}\begin{pmatrix}\dot{\boldsymbol{v}}_\mathrm{g}\\ \dot{\boldsymbol{\omega}}_\mathrm{g}\end{pmatrix} + \begin{pmatrix}0\\ \boldsymbol{\omega}_\mathrm{g}\times\mathbf{I}_\mathrm{g}\boldsymbol{\omega}_\mathrm{g}\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{E} & 0\\ \tilde{\boldsymbol{r}}_\mathrm{gh} & \mathbf{E}\end{pmatrix}\begin{pmatrix}\boldsymbol{f}_\mathrm{h}\\ \boldsymbol{n}_\mathrm{h}\end{pmatrix},\end{aligned} \tag{55.37}$$

where $\mathbf{E}$ is a 3×3 identity matrix, and the symbols with the suffix g indicate the corresponding values observed around the mass centroid of the $n+1$ link system.

In the above equations, (55.8) describes an internal joint motion (termed *local motion*) of the system, whereas (55.37) describes the overall motion (termed *global motion*) about the centroid of the system. As a result of force input $\boldsymbol{F}_\mathrm{h}$, the floating chain induces both the local motion around its articulated joints and the global translation and rotation with respect to the centroid.

From (55.8) and (55.37), the acceleration of the manipulator end tip $\boldsymbol{\alpha}_\mathrm{h}$ can be expressed in the inertial frame as

$$\boldsymbol{\alpha}_\mathrm{h} = \mathbf{G}^*\boldsymbol{F}_\mathrm{h} + \boldsymbol{d}^*\,, \tag{55.38}$$

where

$$\mathbf{G}^* = \hat{\mathbf{J}}\hat{\mathbf{H}}^{-1}\hat{\mathbf{J}}^\mathrm{T} + \mathbf{R}_\mathrm{h}\mathbf{M}^{-1}\mathbf{R}_\mathrm{h}^\mathrm{T}\,, \tag{55.39}$$

$$\mathbf{R}_\mathrm{h} = \begin{pmatrix}\mathbf{E} & -\tilde{\boldsymbol{r}}_\mathrm{gh}\\ 0 & \mathbf{E}\end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} w\mathbf{E} & 0\\ 0 & \mathbf{I}_\mathrm{g}\end{pmatrix}, \tag{55.40}$$

and $\boldsymbol{d}^*$ is a velocity dependent term.

Equations (55.38)–(55.40) are expressions for the motion of the hand (the point at which collision occurs) induced by the impact force $\boldsymbol{F}_\mathrm{h}$, where the matrix $\mathbf{G}^*$, which is the augmented version of (55.10), represents the dynamic characteristics of this system.

Further augmentation for the inverted inertia matrix has been discussed for the case in which the contact duration is not considered to be infinitesimal [55.51].

Now let us assume the case in which two free-floating chains, A and B, with dynamic characteristics $\mathbf{G}_\mathrm{A}^*$ and $\mathbf{G}_\mathrm{B}^*$ collide with each other at their respective hands (end points) and an impact force $\boldsymbol{F}_\mathrm{h}$ is induced by this collision.

The equations of motion at the instance of collision are

$$\mathbf{G}_\mathrm{A}^*\boldsymbol{F}_\mathrm{h} = \begin{pmatrix}\dot{\boldsymbol{v}}_\mathrm{hA}\\ \dot{\boldsymbol{\omega}}_\mathrm{hA}\end{pmatrix} - \boldsymbol{d}_\mathrm{A}^* \tag{55.41}$$

for the chain A, and

$$\mathbf{G}_\mathrm{B}^*(-\boldsymbol{F}_\mathrm{h}) = \begin{pmatrix}\dot{\mathbf{v}}_\mathrm{hB}\\ \dot{\boldsymbol{\omega}}_\mathrm{hB}\end{pmatrix} - \boldsymbol{d}_\mathrm{B}^* \tag{55.42}$$

for the chain B, where the subscripts A and B indicate the label of the chain.

Assuming that $\mathbf{G}_\mathrm{A}^*$ and $\mathbf{G}_\mathrm{B}^*$ remain constant during the infinitesimal contact duration and the velocity-dependent terms $\boldsymbol{d}_\mathrm{A}^*$ and $\boldsymbol{d}_\mathrm{B}^*$ are small and negligible, integration of (55.41) and (55.42) yields

$$\mathbf{G}_\mathrm{A}^*\int_t^{t+\delta t}\boldsymbol{F}_\mathrm{h}\,\mathrm{d}t = \begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}, \tag{55.43}$$

$$\mathbf{G}_\mathrm{B}^*\int_t^{t+\delta t}\boldsymbol{F}_\mathrm{h}\,\mathrm{d}t = \begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}'_\mathrm{hB}\\ \boldsymbol{\omega}'_\mathrm{hB}\end{pmatrix}, \tag{55.44}$$

where $\{'\}$ indicates the velocity after the collision. Integration of $\boldsymbol{F}_\mathrm{h}$

$$\overline{\boldsymbol{F}}_\mathrm{h} = \lim_{\delta t \to 0} \int_{t}^{t+\delta t} \boldsymbol{F}_\mathrm{h}\, \mathrm{d}t \tag{55.45}$$

represents the impulse (force-time product) acting on both chains. Providing that the total momenta of the two systems are strictly conserved before and after the collision, we obtain the following expression from (55.43) and (55.44)

$$\begin{aligned}\left(\mathbf{G}_\mathrm{A}^* + \mathbf{G}_\mathrm{B}^*\right)\overline{\boldsymbol{F}}_\mathrm{h} = &\left\{\begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}'_\mathrm{hB}\\ \boldsymbol{\omega}'_\mathrm{hB}\end{pmatrix}\right\}\\ &+ \left\{\begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}\right\}.\end{aligned} \tag{55.46}$$

In general collision analysis, the coefficient of restitution (elasticity factor) associated with the relative velocities before and after the collision is often employed [55.79]. If we accept this restitution coefficient for 6-DOF linear and angular velocities, the relationship between relative velocities before and after contact becomes

$$\begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}'_\mathrm{hB}\\ \boldsymbol{\omega}'_\mathrm{hB}\end{pmatrix} = \boldsymbol{\epsilon}\left\{\begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}\right\}, \tag{55.47}$$

where

$$\boldsymbol{\epsilon} \equiv \mathrm{diag}(e_1, \ldots, e_6)\,, \quad 0 \le e_i \le 1 \tag{55.48}$$

is the restitution coefficient matrix.

Substituting (55.47) into (55.46), the impulse induced by this collision can be expressed only by the relative velocity of two points before the contact

$$\overline{\boldsymbol{F}}_\mathrm{h} = (\mathbf{E}_6 + \boldsymbol{\epsilon})\mathbf{G}_\Sigma^{*-1}\mathcal{V}_\mathrm{hAB}\,, \tag{55.49}$$

where

$$\mathbf{G}_\Sigma^* = \mathbf{G}_\mathrm{A}^* + \mathbf{G}_\mathrm{B}^*\,, \tag{55.50}$$

$$\mathcal{V}_\mathrm{hAB} = \begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}. \tag{55.51}$$

Using the introduced notation, the magnitude of impulse is expressed as

$$\| \overline{\boldsymbol{F}}_\mathrm{h} \| = \sqrt{(\mathbf{E}_6 + \boldsymbol{\epsilon})^\mathrm{T}\mathcal{V}_\mathrm{hAB}^\mathrm{T}\mathbf{G}_\Sigma^{*-T}\mathbf{G}_\Sigma^{*-1}\mathcal{V}_\mathrm{hAB}(\mathbf{E}_6 + \boldsymbol{\epsilon})}\,, \tag{55.52}$$

and the velocity after collision becomes

$$\begin{aligned}\begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} = &\left(\mathbf{G}_\mathrm{A}^{*-1} + \mathbf{G}_\mathrm{B}^{*-1}\right)^{-1}\\ &\times\left[(\mathbf{E}_6 + \boldsymbol{\epsilon})\mathbf{G}_\mathrm{B}^{*-1}\begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix}\right.\\ &\left.+(\mathbf{G}_\mathrm{A}^{*-1} - \boldsymbol{\epsilon}\mathbf{G}_\mathrm{B}^{*-1})\begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}\right],\end{aligned} \tag{55.53}$$

where suffixes A and B are interchangeable.

These expressions are considered to be an augmentation of the impact theory for a two-point-mass system into articulated body systems.

Impedance control is a concept by which we can control the manipulator end tip so as to obtain the desired mechanical impedance characteristics. Such control is useful to alter the dynamic characteristics of the arm during the contact phase. In a special case, the desired impedance of the manipulator end tip (hand) may be tuned to achieve *impedance matching* with the colliding target object so that the hand can easily maintain stable contact with the target [55.80].

Let $\mathbf{M}_\mathrm{d}$, $\mathbf{D}_\mathrm{d}$, and $\mathbf{K}_\mathrm{d}$ be the matrices for the desired impedance properties of inertia, viscosity, and stiffness, respectively, measured at the manipulator end point. The equation of motion for the desired system is then expressed as

$$\mathbf{M}_\mathrm{d}\ddot{\boldsymbol{x}}_\mathrm{h} + \mathbf{D}_\mathrm{d}\Delta\dot{\boldsymbol{x}}_\mathrm{h} + \mathbf{K}_\mathrm{d}\Delta\boldsymbol{x}_\mathrm{h} = \boldsymbol{F}_\mathrm{h}\,. \tag{55.54}$$

From (55.8) and (55.54), the impedance control law for a free-floating manipulator system is obtained as [55.81]

$$\begin{aligned}\boldsymbol{\tau}_\mathrm{h} = &\mathbf{H}^*\hat{\mathbf{J}}^{-1}\{\mathbf{M}_\mathrm{d}^{-1}(\mathbf{D}_\mathrm{d}\Delta\dot{\boldsymbol{x}}_\mathrm{h} + \mathbf{K}_\mathrm{d}\Delta\boldsymbol{x}_\mathrm{h} - \boldsymbol{F}_\mathrm{h})\\ &- \dot{\hat{\mathbf{J}}}\dot{\boldsymbol{\phi}} - \ddot{\boldsymbol{x}}_\mathrm{gh}\} - \hat{\mathbf{J}}^\mathrm{T}\boldsymbol{F}_\mathrm{h} + \boldsymbol{c}^*\,.\end{aligned} \tag{55.55}$$

The usefulness of the impedance control in free-flying space robots has been discussed in [55.81–83].

55.3.12 Dynamics of Mobile Robots

The equation of motion for a mobile robot that has multiple articulated limbs, such as that shown in Fig. 55.29,

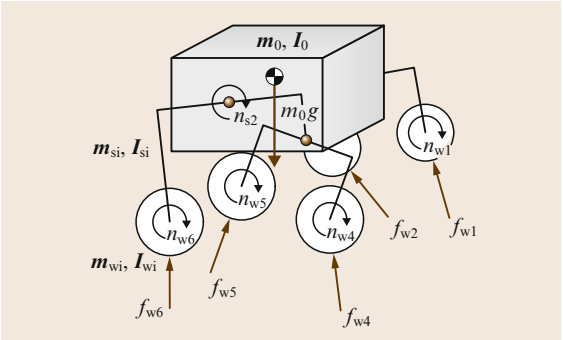


Fig. 55.29 Schematic model of a mobile robot

is given by the following equation

$$\begin{pmatrix} \mathbf{H}_{\mathrm{b}} & \mathbf{H}_{\mathrm{bm}1} & \cdots & \mathbf{H}_{\mathrm{bm}k} \\ \mathbf{H}_{\mathrm{bm}1}^{\mathrm{T}} & \mathbf{H}_{\mathrm{m}11} & \cdots & \mathbf{H}_{\mathrm{m}1k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_{\mathrm{bm}k}^{\mathrm{T}} & \mathbf{H}_{\mathrm{m}1k}^{\mathrm{T}} & \cdots & \mathbf{H}_{\mathrm{m}k k} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{x}}_{\mathrm{b}} \\ \ddot{\boldsymbol{\phi}}_1 \\ \vdots \\ \ddot{\boldsymbol{\phi}}_k \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_{\mathrm{b}} \\ \boldsymbol{c}_{\mathrm{m}1} \\ \vdots \\ \boldsymbol{c}_{\mathrm{m}k} \end{pmatrix} = \begin{pmatrix} \boldsymbol{F}_{\mathrm{b}} \\ \boldsymbol{\tau}_1 \\ \vdots \\ \boldsymbol{\tau}_n \end{pmatrix} + \begin{pmatrix} \mathbf{J}_{\mathrm{b}}^{\mathrm{T}} \boldsymbol{F}_{\mathrm{ex}} \\ \mathbf{J}_{\mathrm{m}1}^{\mathrm{T}} \boldsymbol{F}_{\mathrm{ex}1} \\ \vdots \\ \mathbf{J}_{\mathrm{m}k}^{\mathrm{T}} \boldsymbol{F}_{\mathrm{ex}k} \end{pmatrix}, \tag{55.56}$$

where the symbols have the following meanings:

k number of limbs
$\boldsymbol{x}_{\mathrm{b}} \in \mathbb{R}^6$: position/orientation of the base
$\boldsymbol{\phi} = (\boldsymbol{\phi}_1^{\mathrm{T}}, \ldots, \boldsymbol{\phi}_k^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^n$: articulated joint angles
$\boldsymbol{x}_{\mathrm{ex}} = (\boldsymbol{x}_{\mathrm{ex}1}^{\mathrm{T}}, \ldots, \boldsymbol{x}_{\mathrm{ex}k}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{6k}$: position/orientation of the end points
$\boldsymbol{F}_{\mathrm{b}} \in \mathbb{R}^6$: forces/moments directly apply on the base
$\boldsymbol{\tau} = (\boldsymbol{\tau}_1^{\mathrm{T}}, \ldots, \boldsymbol{\tau}_k^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^n$: joint articulated torque
$\boldsymbol{F}_{\mathrm{ex}} = (\boldsymbol{F}_{\mathrm{ex}1}^{\mathrm{T}}, \cdots, \boldsymbol{F}_{\mathrm{ex}k}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{6k}$: external forces/moments on the end points.

Note that for Fig. 55.29, $\boldsymbol{F}_{\mathrm{ex}i} = [\boldsymbol{f}_{\mathrm{w}i}^{\mathrm{T}}, \boldsymbol{n}_{\mathrm{w}i}^{\mathrm{T}}]^{\mathrm{T}}$.

Comparing the above equations with (55.1), no difference is observed in the mathematical structure. The gravity force on the vehicle main body and the configuration-dependent gravity terms of the articulated bodies are included in $\boldsymbol{c}_{\mathrm{b}}$ and $\boldsymbol{c}_{\mathrm{m}i}$, respectively. In practice, however, one substantial difference is the existence of ground contact forces/moments at the end point of each limb. Unlike floating target capture discussed in Sect. 55.3.10, the contact is not considered impulsive, but instead continues for a nonnegligible period of time. In such cases, a well-accepted approach is to explicitly evaluate the contact forces/moments $\boldsymbol{F}_{\mathrm{ex}}$, according to the virtual penetration of the end point into the collided object or the ground surface [55.79].

In cases in which each limb has a wheel on its end terminal, rather than the point penetration model, a wheel traction model will be adopted to evaluate $\boldsymbol{F}_{\mathrm{ex}}$. For planetary exploration missions, rovers (mobile robots) are expected to travel over natural rough terrain. A number of studies have examined the modeling of tire traction forces on loose soil, called *regolith* (where there is no organic component) [55.84–98]. Particularly, these studies investigate the soil mechanics called *terramechanics* to understand the tractive forces generated by wheels.

In the following subsection, models for wheel traction mechanics are summarized.

55.3.13 Wheel Traction Mechanics

Terramechanics is the study of soil properties, specifically the interaction of wheeled, legged, or tracked vehicles and various surfaces. For the modeling of the wheel traction forces and the analysis of the vehicle mobility, textbooks written by *Bekker* [55.29, 30] and *Wong* [55.99] are good references. Although these books were written in the 1960s and 1970s, basic formulae from these books are frequently cited by researchers even today [55.88]. In this subsection, the models for a rigid wheel on loose soil are summarized.

Slip Ratio and Slip Angle

Slips are generally observed when a rover travels on loose soil. In particular, slips in the lateral direction are observed during steering or slope-traversing maneuvers.

The slip in the longitudinal direction is measured by slip ratio s, which is defined as a function of the longitudinal traveling velocity v_x of the wheel and the circumference velocity of the wheel $r\omega$ (r is the wheel radius and ω is the angular velocity of the wheel)

$$s = \begin{cases} \dfrac{r\omega - v_x}{r\omega} & \text{if } |r\omega| > |v_x| : \text{driving} \\ \dfrac{r\omega - v_x}{v_x} & \text{if} |r\omega| < |v_x| : \text{braking} . \end{cases} \tag{55.57}$$

The slip ratio takes a value from -1 to 1.

On the other hand, the slip in the lateral direction is measured by slip angle β, which is defined in terms of v_x and the lateral traveling velocity v_y as

$$\beta = \tan^{-1}\left(\frac{v_y}{v_x}\right) . \tag{55.58}$$

Note that the above definitions, (55.57) and (55.58) have been traditionally used in the vehicle community as standards. However, planetary rovers in a challenging terrain, such as Spirit and Opportunity on Mars, experienced the cases in which the rovers slip backward while attempting to drive up hill, or travel faster than the wheel's circumference velocity in downhill driving. In these cases, the slip ratio can exceed the range from -1 to 1. Also while traversing side slopes, the case may arise in which $v_y > 0$ but v_x is nearly 0, making the definition (55.58) nearly singular. Therefore, these definitions are needed to be further discussed for the rovers in very loose terrain.

Wheel–Soil Contact Angle

Figure 55.30 depicts a schematic model of a rigid wheel contacting loose soil. In the figure, the angle from the surface normal to the point at which the wheel initially makes contact with the soil ($\angle$ AOB) is defined as the

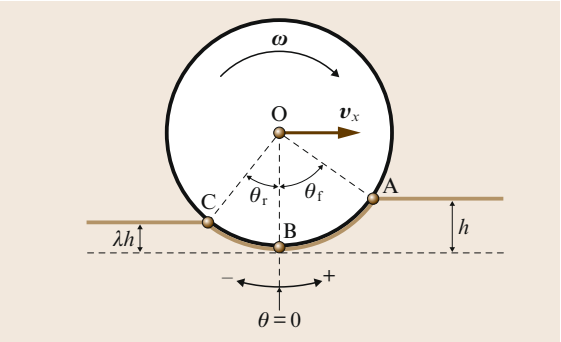


Fig. 55.30 Wheel contact angles

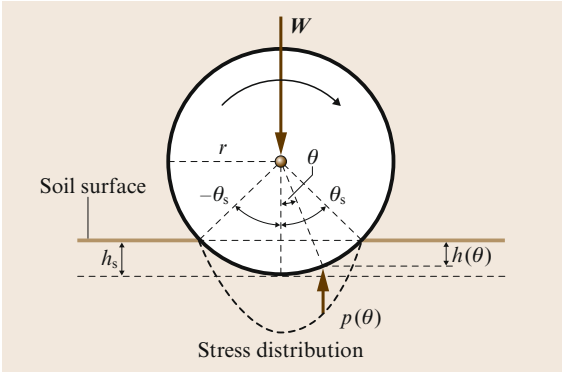


Fig. 55.31 Static sinkage

entry angle. The angle from the surface normal to the point at which the wheel departs from the soil (∠ BOC in Fig. 55.30) is the exit angle. The wheel contact region on loose soil is represented from the entry angle to the exit angle.

The entry angle θ_f is geometrically described in terms of wheel sinkage h as

$$\theta_f = \cos^{-1}\left(1 - \frac{h}{r}\right) . \tag{55.59}$$

The exit angle θ_r is described using the wheel sinkage ratio λ, which denotes the ratio between the forward and rear sinkage of the wheel

$$\theta_r = \cos^{-1}\left(1 - \frac{\lambda h}{r}\right) . \tag{55.60}$$

The value of λ depends on the soil characteristics, the wheel surface pattern, and the slip ratio. It becomes smaller than 1.0 when the soil compaction occurs, but can be greater than 1.0 when the soil is dug up by the wheel and transported to the rear region of the wheel.

Wheel Sinkage

The amount of wheel sinkage is constituted by static and dynamic components. The static sinkage depends on the vertical load on the wheel, while the dynamic sinkage is caused by the rotation of the wheel.

According to the equation formulated by *Bekker* [55.29], the static stress $p(h)$ generated under a flat plate, which has a sinkage h and a width b, is calculated as

$$p(h) = \left(\frac{k_c}{b} + k_\phi\right) h^n , \tag{55.61}$$

where k_c and k_ϕ are pressure-sinkage modules, and n is the sinkage exponent. Applying (55.61) to the wheel, as shown in Fig. 55.31, the static sinkage is evaluated as follows.

First, the wheel sinkage $h(\theta)$ at an arbitrary wheel angle θ is geometrically given by

$$h(\theta) = r(\cos\theta - \cos\theta_s) , \tag{55.62}$$

where θ_s is the static contact angle. Then, substituting (55.62) into (55.61) yields

$$p(\theta) = r^n \left(\frac{k_c}{b} + k_\phi\right) (\cos\theta - \cos\theta_s)^n . \tag{55.63}$$

The wheel eventually sinks into the soil until the stress from the soil balances the vertical load W on the wheel.

$$\begin{aligned} W &= \int_{-\theta_s}^{\theta_s} p(\theta) br \cos\theta \, d\theta \\ &= r^{n+1} (k_c + k_\phi b) \int_{-\theta_s}^{\theta_s} (\cos\theta - \cos\theta_s)^n \cos\theta \, d\theta . \end{aligned} \tag{55.64}$$

Using this equation, the static contact angle θ_s is evaluated for the given W. In practice, (55.64) does not yield a closed form solution for θ_s, although θ_s can be evaluated numerically.

Finally, the static sinkage h_s is obtained by substituting θ_s into the following equation

$$h_s = r(1 - \cos\theta_s) . \tag{55.65}$$

However, as illustrated in Fig. 55.32, the dynamic sinkage becomes a complicated function depending on the slip ratio of the wheel, the wheel surface pattern, and the soil characteristics. Although it is difficult to obtain an analytical form for the dynamic sinkage, it is again possible to evaluate the dynamic sinkage numerically, using the condition $W = F_z$, where F_z is the normal force given by (55.76), which will be presented

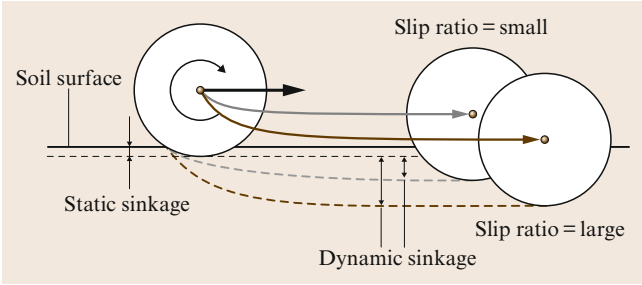


Fig. 55.32 Dynamic sinkage

later herein. The force F_z increases with the wheel sinkage because the area of the contact patch increases accordingly.

Stress Distribution Under the Wheel

Based on terramechanics models, the stress distribution under the rotating wheel can be modeled as shown in Fig. 55.33.

The normal stress $\sigma(\theta)$ is determined by the following equation [55.87, 88]

$$\sigma(\theta) = r^n \left(\frac{k_c}{b} + k_\phi\right)(\cos\theta - \cos\theta_f)^n \quad \text{for } \theta_m \le \theta < \theta_f ,$$

$$\sigma(\theta) = r^n \left(\frac{k_c}{b} + k_\phi\right)\left\{\cos\left[\theta_f - \frac{\theta - \theta_r}{\theta_m - \theta_r}(\theta_f - \theta_m)\right] - \cos\theta_f\right\}^n \quad \text{for } \theta_r < \theta \le \theta_m . \tag{55.66}$$

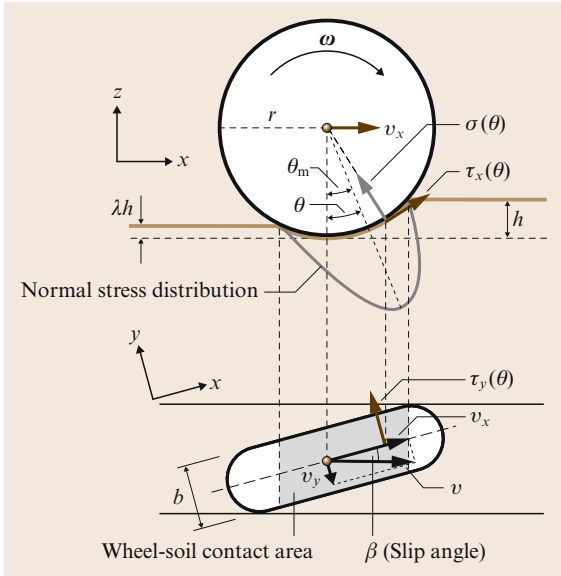


Fig. 55.33 Stress distribution model under a wheel

Note that the above equations are based on Bekker's formula, as given in (55.61), and they become equivalent to the *Wong–Reece* model for normal stress [55.100] when $n = 1$. Also note that by linearizing this distribution, *Iagnemma* et al. [55.84, 88] developed a Kalman-filter-based method to estimate the soil parameters.

The term θ_m is the specific wheel angle at which the normal stress is maximized

$$\theta_m = (a_0 + a_1 s)\theta_f , \tag{55.67}$$

where a_0 and a_1 are parameters that depend on the wheel–soil interaction. Their values are generally assumed to be $a_0 \approx 0.4$ and $0 \le a_1 \le 0.3$ [55.100].

The maximum terrain shear force is a function of the terrain cohesion c and internal friction angle ϕ, and can be computed from Coulomb's equation

$$\tau_{max}(\theta) = c + \sigma_{max}(\theta)\tan\phi . \tag{55.68}$$

Based on the above equation, the shear stresses under the rotating wheel, $\tau_x(\theta)$ and $\tau_y(\theta)$, are written as follows [55.101]

$$\tau_x(\theta) = [c + \sigma(\theta)\tan\phi]\left(1 - e^{-j_x(\theta)/k_x}\right) , \tag{55.69}$$

$$\tau_y(\theta) = [c + \sigma(\theta)\tan\phi]\left(1 - e^{-j_y(\theta)/k_y}\right) , \tag{55.70}$$

where k_x and k_y are the shear deformation moduli in each direction. In addition, j_x and j_y, which are the soil deformations in each direction, can be formulated as a function of the wheel angle θ with the slip ratio and the slip angle, respectively [55.89, 100]

$$j_x(\theta) = r[\theta_f - \theta - (1 - s)(\sin\theta_f - \sin\theta)] , \tag{55.71}$$

$$j_y(\theta) = r(1 - s)(\theta_f - \theta)\tan\beta . \tag{55.72}$$

Drawbar Pull: F_x

Using the normal stress $\sigma(\theta)$ and the shear stress in the x direction $\tau_x(\theta)$, the drawbar pull F_x, which is the net traction force exerted from the soil to the wheel, is calculated as the integral from the entry angle θ_f to the exit angle θ_r [55.100]

$$F_x = rb\int_{\theta_r}^{\theta_f} [\tau_x(\theta)\cos\theta - \sigma(\theta)\sin\theta]\, d\theta . \tag{55.73}$$

Side Force: F_y

The side force F_y appears in the lateral direction of the wheel when the vehicle makes steering maneuvers or traverses a side slope. The side force is decomposed into two components [55.89]

$$F_y = F_u + F_s ,$$

where F_u is the force produced by the shear stress in the y direction $\tau_y(\theta)$ underneath the wheel and F_s is the reaction force generated by the bulldozing phenomenon on a side face of the wheel. The above equation can be rewritten as

$$F_y = \int_{\theta_r}^{\theta_f} \left\{ \underbrace{rb\tau_y(\theta)}_{F_u} + \underbrace{R_b[r - h(\theta)\cos\theta]}_{F_s} \right\} d\theta \,. \tag{55.74}$$

Here, *Hegedus's bulldozing resistance estimation* [55.102] is employed to evaluate the side face force F_s. As shown in Fig. 55.34, a bulldozing resistance R_b is generated on a unit width blade when the blade moves toward the soil. According to Hegedus's theory, the bulldozed area is defined by a destructive phase that is modeled by a planar surface. In the case of a horizontally placed wheel, the angle of approach α' should be zero; R_b can then be calculated as a function of wheel sinkage $h(\theta)$

$$R_b(h) = D_1 \left[ch(\theta) + D_2 \frac{\rho_d h^2(\theta)}{2} \right] , \tag{55.75}$$

where

$$D_1(X_c, \phi) = \cot X_c + \tan(X_c + \phi) \,,$$

$$D_2(X_c, \phi) = \cot X_c + \frac{\cot^2 X_c}{\cot \phi} \,.$$

In the above equations, ρ_d denotes the soil density. Based on *Bekker*'s theory [55.29], the destructive angle X_c can be approximated as

$$X_c = 45^\circ - \frac{\phi}{2} \,.$$

Normal Force: F_z

The normal force F_z is obtained in the same manner as for the (55.73) [55.100]

$$F_z = rb \int_{\theta_r}^{\theta_f} [\tau_x(\theta)\sin\theta + \sigma(\theta)\cos\theta] \, d\theta \,, \tag{55.76}$$

which should balance the normal load of the wheel in a static condition.

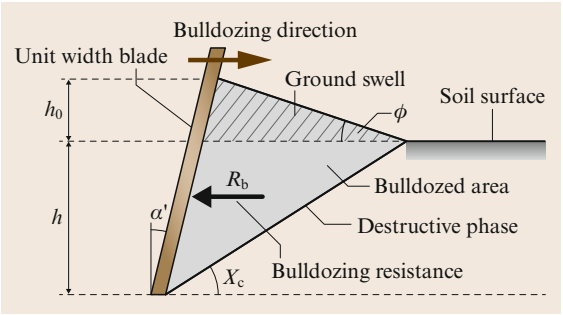


Fig. 55.34 Estimation model of the bulldozing resistance

Motion dynamics simulation for a vehicle traveling over loose soil can be performed by plugging the forces F_x, F_y, and F_z obtained from the above equations into the equation of motion (55.5).

A better understanding of the soil–wheel contact and traction mechanics is important in order to improve the navigation and control behavior of exploration rovers, in terms of minimization of wheel slippage, for example. Reducing the wheel slippage will increase the power efficiency of surface locomotion, decrease the errors in path tracking maneuvers, and decrease the risks of wheel spinning and sinking, which can cause immobilization of the vehicle.

One key in realizing such advanced control of slippage minimization is determining how to properly estimate the slip ratios and slip angles in real time using on-board sensors. The slip ratio is determined by the ratio between the wheel spinning velocity and the traveling velocity of the vehicle, but proper sensing of the velocity of the vehicle is usually difficult. One simple solution is to use a free wheel specialized for traveling velocity measurement. Another solution is to employ inertial sensors, which are however usually subject to noise and drift.

An alternative, but promising possibility is visual odometry, which is based on optical flow or feature tracking in the sequence of optical images. Actually, this technique has been applied to the Mars exploration rovers, Spirit, and Opportunity, in their long range navigation, and verified to be very useful. Particularly, the algorithm based on feature detection and tracking using stereo pair of cameras provides reliable results with good accuracy for the estimation of driving distance as well as the wheel slippage [55.28].

Avidbots Ex1016
Page 1473 of 2154

55.4 Future Directions of Orbital and Surface Robotic Systems

55.4.1 Robotic Maintenance and Service Missions

For many years, we have sent satellites and other systems into space without caring too much about what might happen at the end of their life cycle. There has been recently awareness about the dramatic increase of space debris and the danger of a fatal chain reaction increased in collisions. Generally speaking, space debris removal may become a prerequisite for future spaceflight. Space systems above approx. 600 km flight path altitude are not currently accessible to astronauts by means of present transport systems and therefore are excluded from any kind of human removal, repair, or maintenance.

In contrast, satellites equipped with robotic arms or humanoid robonauts may be remotely controlled or only supervised from Earth in any orbit including the geostationary one. In the future, they should be able to support astronauts during routine and maintenance work on space stations, capture uncontrollably tumbling satellites, prolong their life-time by repair or refuel, deorbit or relocate them if necessary. Efficient telerobotic and telepresence technologies allow us to select flexibly the appropriate level of robot autonomy within *shared autonomy* frameworks between ground operator and space robot.

Telepresence technologies as mentioned above make sure that by real-time feedback of stereo images and force/torque information, the operator on the ground gets the feeling as though he was actually working at the remote site. High-quality telepresence requires low round-trip communication time delays. The challenge here is twofold: (a) to provide the mentioned technically feasible communication infrastructure, and (b) to apply the mentioned optimized delay-compensating telepresence technologies which yield satisfactory haptic feedback up to 650 ms delay. For large robots mounted on a carrier satellite, their dynamic interactions, including the physical contacts when grasping a target, have to be mastered (Sect. 55.3). Autonomous skills are needed for supporting a human ground operator in performing the risky task of grasping and stabilizing noncooperative tumbling targets (satellite or space debris).

Due to the high cost of space validation missions, simulation capabilities are crucial for the development and verification of on-orbit servicing systems. This applies both for the required hardware simulation facilities including sensors and illumination effects as well as for dynamics modeling techniques and software tools. Various hardware simulators using industrial robots for simulation of chaser and target satellite motion and the dynamic interaction with the space-robot have been built up not only in DLR (Fig. 55.35).

Robotic maintenance and service missions for space infrastructure have been a long-term dream in the space robotics community since their conceptual designs were first published in ARAMIS report in the early 1980s (Fig. 55.1) [55.1].

ROTEX, ETS-VII, Ranger, and ASTRO that were introduced in the earlier section are technological developments toward this goal, but robotic maintenance and service missions have not become routinely operational yet (A good comparative study of orbital robotic missions is provided by [55.103]). The Hubble space telescope (HST) is a huge space telescope which has the capability to be serviced in orbit, but it has been visited by Space Shuttle and serviced (components exchanged and trouble fixed) only by human EVA. After the COLUMBIA accident in 2003, NASA seriously considered the possibility of robotic maintenance of the HST, investigating available technologies and selecting a prime contractor of the mission development. Figure 55.36 depicts one possible configuration for the robotic rescue mission. Ultimately, it was decided to perform this last servicing mission with human astronauts. Maintenance of the HST involves tasks that are too complicated to be done by a robot, because the HST itself was designed for human-based maintenance and not specifically designed for robots.

Robonaut, which is described in the following subsection, is therefore considered as an interesting option to be capable in conducting practical maintenance and service missions, due to its compatibility and similar level of dexterity with human astronauts.



Fig. 55.35 OOS-SIM, a hardware-in-the-loop simulator in DLR



Fig. 55.36 A conceptual drawing for robotic rescue of Hubble space telescope

55.4.2 Robonaut and JUSTIN

Robonaut is a humanoid robot designed by the Robot Systems Technology Branch at NASA Johnson Space Center in a collaborative effort with DARPA. The Robonaut project seeks to develop and demonstrate a robotic system that can function as an EVA astronaut equivalent. Robonaut jumps generations ahead by eliminating the robotic scars (e.g., special robotic grapples and targets), but it still keeps the human operator in the control loop through its telepresence control system. Robonaut is designed to be used for *EVA* tasks (extravehicular activities or space walks), i. e., those which were not specifically designed for robots.

A key challenge is to build machines that can help humans work and explore in space. Working side by side with humans, or going where the risks are too great for people, machines like Robonaut will expand capabilities for construction and discovery. Over the past five decades, space flight hardware has been designed for human servicing. Space walks are planned for most of the assembly missions for the ISS, and they are a key contingency for resolving on-orbit failures. To maintain compatibility with existing EVA tools and equipments, a humanoid shape and an assumed level of human performance (at least a human in a space suit) are required for this robotic surrogate.

The manipulator and dexterous hand have been developed with a substantial investment in mechatronics design. The arm structure has embedded avionics elements within each link, reducing cabling and noise interference. Robonaut has been designed based on a biologically inspired approach. For example, it uses a chordate neurological system in data management, bringing all feedback to a central nervous system, where even low-level servo control is performed. Such a biologically inspired approach is extended to left-right computational symmetry, sensor and power duality and kinematical redundancy, enabling learning and optimization in mechanical, electrical and software forms.

Robonaut has a broad mix of sensors including thermal, position, tactile, force and torque instrumentation, with over 150 sensors per arm. The control system for Robonaut includes an onboard, real-time CPU with miniature data acquisition and power management. Offboard guidance is delivered with human supervision using a telepresence control station with human tracking.

Robonaut 2 (Fig. 55.37), the latest generation of the Robonaut family, launched to the ISS aboard Space Shuttle Discovery on the STS-133 mission in February 2011. It is the first humanoid robot in space, and although its initial job is demonstrating its capabilities inside the space station, the goal is that through upgrades and advancements it will one day venture outside the station to help spacewalkers make repairs or additions to the station. Robonaut 2 is a dexterous, anthropomorphic robotic torso that has significant technical improvements over its predecessor to make it a far more valuable tool for astronauts. Upgrades include: increased force sensing, greater range of motion, higher bandwidth, and improved dexterity. Robonaut 2's integrated mechatronic design results in a more compact and robust distributed control system with a fraction of the wiring of the original Robonaut.

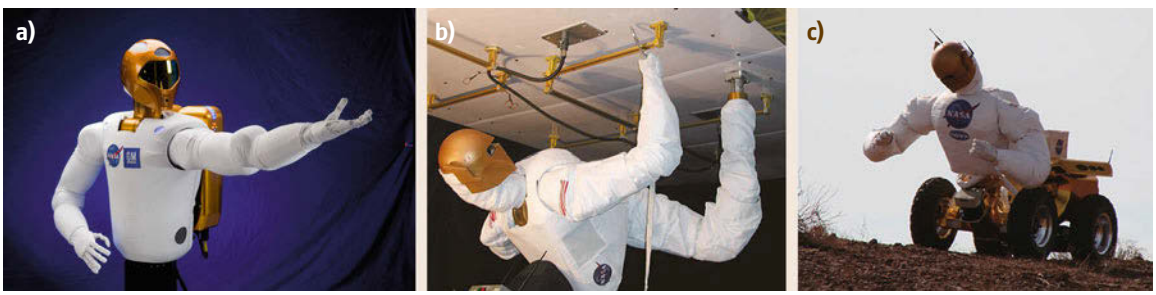


Fig.55.37a–c NASA's Robonaut family: **(a)** Robonaut 2, **(b)** Zero-G Leg for surface inspection of ISS, **(c)** Centaur with a surface mobility system

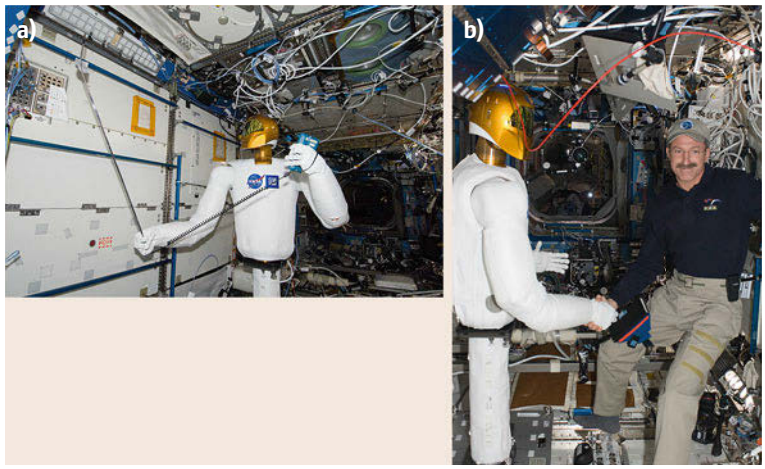


Fig.55.38a,b Robonaut 2 onboard ISS: **(a)** measuring airflow, **(b)** shaking hands with ISS Commander Dan Burbank

Robonaut 2, also called R2, has completed many firsts during its two years on the ISS. During its initial checkout, it used American sign language to say *Hello World.* R2 illustrated its unique control system design that permits it to work directly with astronauts by shaking hands with ISS Commander Dan Burbank (Fig. 55.38). More recently, it has been using standard crew tools to measure airflow and demonstrate its ability to perform autonomous inventory scans. As part of gaining experience that will be useful once R2 starts working on the outside of the Space Station, on-board crew have successfully demonstrated teleoperation. Using a variety of sensors that track human hand, arm and neck motion, astronaut Tom Marshburn, while also onboard the station, became the first person to remotely control R2 to have it catch a free flying object inside the ISS.

One potential application of Robonaut technology is a regular monitoring and contingent maintenance work of human habitant modules of the space station. Figure 55.37b depicts such an application where Robonaut crawls on the surface of the station module by using hand rails which were originally designed for human EVA.

The application of the humanoid robot is not limited to orbital tasks. Figure 55.37c depict an idea to combine the humanoid torso on a surface mobility system, which shall be useful for robotic planetary explorations.

DLR's anthropomorphic JUSTIN is based on high-fidelity joint-torque-controlled light weight-technology and adjustable whole-body compliance in Cartesian space. JUSTIN on the mobile platform (Fig. 55.39) has actuated joints and torque controlled sensors. With JUSTIN's upper body the new delay compensating technologies have been verified using copies of JUSTIN's light weight arms as force reflecting hand-controllers up to delays of slightly more than 700 ms.

The European space agency ESA too, is pushing forward robonaut-type concepts, e.g., via testing a dexterous 4-finger-hand DEXHAND as developed in contract by DLR (Fig. 55.40) In spring 2012, the DEXHAND successfully passed the acceptance test and is delivered to ESA.

55.4.3 Aerial Platforms

There are three planetary candidates for aerial robotic systems: Venus, Mars, and Titan (a moon of Saturn) [55.104, 105]. Venus has a very dense but hot atmosphere (460 °C and 65 kg m^{-3} at the surface), and so can easily float relatively heavy payloads. Mars has a very thin and cold atmosphere (somewhat variable but often −100 °C and 0.02 kg m^{-3}). Titan has an atmosphere even colder than Mars ($\approx$ 100 K) but about 50% denser than Earth's atmosphere. Thus very different vehicles have been envisioned for the three candidate mission targets. On Venus, buoyant devices are generally considered, especially those that can continuously or periodically rise high enough to reach moderate temperatures where conventional electronics can survive. One candidate is to use a phase-change fluid as part of the buoyant system, so that the fluid can condense in the cool upper atmosphere and be trapped in a pressure vessel, causing a loss of buoyancy and allowing the vehicle to descend, possibly all the way to the surface. After a brief stay, and before the heat flux to the interior of the device destroys all the sensitive equipment, a valve would be opened so that the phase change fluid can evaporate, increasing the buoyancy and allowing the craft to ascend to the cool upper atmosphere. After a suitable period of heat rejection into this cool zone, the process can be repeated, perhaps indefinitely. The density of the Venus atmosphere is sufficiently high that powered dirigibles can be used, so that the buoy-

ant vehicles can use propulsion and steering to reach particular locations in the atmosphere or on the surface [55.106].

In contrast, the Mars atmosphere is too thin for powered dirigibles to work (at least with the power-to-weight ratio of any current propulsion technology). Balloon aerobots could be deployed in the Mars atmosphere, and could ascend and descend, but probably could not be steered precisely to specific locations, at least not by use of a propulsion system. Polar balloons could circumnavigate either pole many times, or equatorial balloons could make one partial circuit around the planet, until they impact the *Tharsis Bulge*, a North–South string of high-altitude volcanoes that represents an essentially impenetrable barrier to any equatorial balloon having a reasonable payload. Because of the problems with *lighter-than-air* vehicles in the thin Mars atmosphere, there has been considerable study of airplanes for use in exploring Mars. Aircraft can be designed to have reasonable lift-to-drag ratios in the Mars atmosphere, so that their performance is not too different from airplanes on Earth. Most often considered are gliders that deploy directly from an aeroshell that comes in to the Mars atmosphere at hypersonic velocity, and then proceed to glide hundreds or a thousand kilometers before impact. One common mission concept is to fly down the great Valles Marineris canyon, taking high-resolution imagery and spectrometry of the walls of that canyon. Powered aircraft have also been considered, including those that land and regenerate their propellant (e.g., using solar power and atmospheric CO_2) so as to be able to make multiple flights.

On Titan, like Venus, buoyant devices are generally considered more attractive than surface vehicles (although helicopters have been proposed). Also like Venus, the atmosphere on Titan contains many obscuring particles and aerosols so that high-resolution imaging over a broad spectrum is only possible by getting close to the surface. This makes balloons or powered dirigibles very attractive. On Venus the extreme surface temperature makes it challenging to make a surface vehicle operate for any extended duration. On Titan, there is a significant risk that some sort of hydrocarbon *goo* exists on the surface that might foul any surface vehicle. Thus both Titan and Venus are considered especially attractive targets for the use of aerobots, especially in the form of powered dirigibles. Navigation of such aerobots presumably would be accomplished primarily by sensing the terrain and navigating relative to any landmarks that can be discerned. When these vehicles operate in the upper atmosphere, they can augment their position knowledge by means of sun or star tracking (as referenced to the local vertical). Deeper in the atmosphere, this may not be possible. One key issue is whether direct communications to Earth are envisioned, or relay via satellite. If there is a satellite in orbit, it can provide considerable radio-navigation assistance and relatively frequent communications when the aerobot is on the side away from the Earth (both Venus and Titan spin very slowly). But a satellite relay is expensive, so the least expensive options require that the dirigible have a large high-gain antenna (usually presumed to be inside the gas bag). Radio-based servo pointing at the Earth will provide precise navigation information (again along with precise measurements of local vertical). However, when the aerobot goes out-of-sight beyond the limb of the planet, it may spend days or weeks out of communications with the Earth. This is probably the situation calling for the highest degree of autonomy of any that have been envisioned in robotic planetary exploration of the solar system.

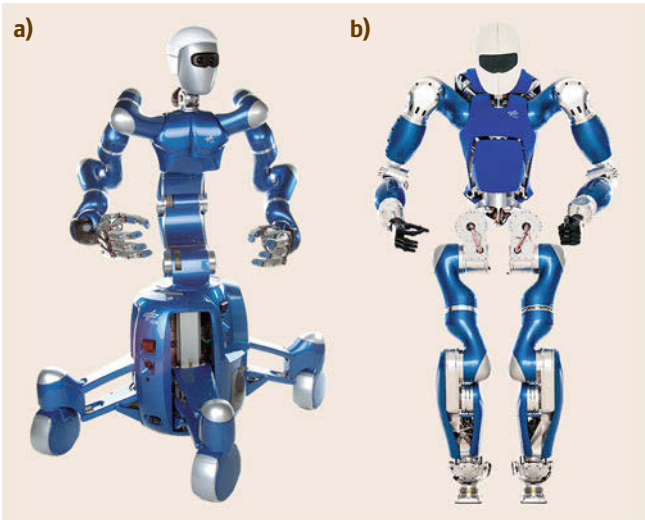


Fig.55.39a,b DLR' JUSTIN, wheeled version **(a)** and legged version **(b)**



Fig. 55.40 DEXHAND

Avidbots Ex1016
Page 1477 of 2154

55.4.4 Mobility Concepts and Subsurface Platforms

For high mobility on Moon, planets, and asteroids there is still not a final answer which technology would be optimal. Although multilegged crawlers (e.g., DLR's six-legged version as shown in Fig. 55.41) seem to be the best alternative for investigating steep craters, 4-wheeled rovers may climb up and down unbelievably steep slopes. May be wheel–leg combinations as realized in JPL's ATHLETE (Fig. 55.42) or in DLR's conceptual design (Fig. 55.43) will turn out to be the optimal solution.

Precise autonomous landing based on visual data is a prerequisite for exploration, closely related to the Future Space Systems program.

DLR's main interest however aims at fast locomotion by local autonomy thus (including collision avoidance and real-time path planning) circumventing the problem of long signal delays from 3 s (Moon) to 15–30 min (Mars). Stereo cameras with field-programmable gate array (FPGA) processor chips are capable of modeling the environment in 3-D real time, using e.g., the so-called semiglobal matching (SGM). Thus the goal of moving up to 10 km per hour seems realizable now.

Other modes of mobility may be superior when gravity is e.g., only 10 000 times smaller than that on earth as is the case on some asteroids. e.g., for a Japanese mission Hayabusa 2 a *jumping shoe-box* is developed by DLR using just a small excentric motor that causes moderate hopping motions over a few 100 m without reaching the fairly low escape velocity.

Subsurface exploration of planetary bodies holds great promise: it is believed that a liquid-water aquifer may exist at significant depths on Mars, and perhaps an under-ice ocean on Europa and Ganymede which probably represent the best possible locations within the solar system to look for extant (as opposed to extinct) extraterrestrial life. Also, in the lunar polar *dark craters* there is some evidence of the existence of water ice or other volatiles, and perhaps there exists a layered geologic record of impacts in the Earth–Moon system in these cold-traps. Even access to a depth of a few meters holds the promise of reaching pristine scientific samples that have not been exposed to thermal cycling or ionizing radiation [55.107].

The prevailing wisdom has been that traditional sorts of drilling rigs are required to access deep underground, involving drill towers, multisegmented drill strings, large robotic systems to serve the function of a terrestrial drilling crew, and large power systems. Also, terrestrial drilling is usually done using large amounts of fluids (water, air, or mud) to flush away cuttings and to cool and lubricate the cutter. The NASA Mars Technology Program has funded contractors that demonstrated reaching 10 m of depth in a realistic setting with segmented drill strings without the use of fluids. While this is much less than needed to reach the putative liquid water, it is much more than is reachable by previous techniques [55.108].



Fig. 55.41 DLR's six-legged crawler

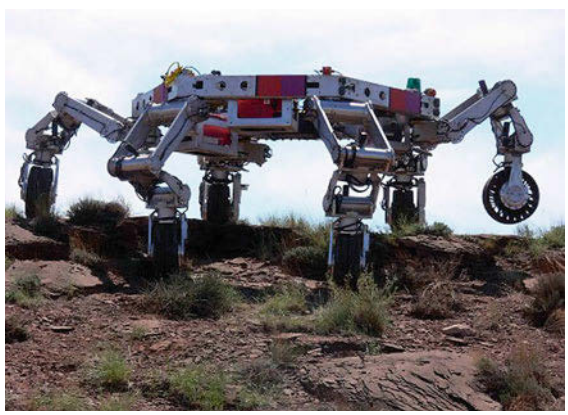


Fig. 55.42 JPL's ATHLETE



Fig. 55.43 Modular rover concept

Other approaches have been proposed such as Moles or Inchworms that could be relatively self-contained and yet might reach great depths without the mass and complexity of a large drill tower and segmented drill string. A key issue is that it appears that the needed energy cannot be stored on-board such self-contained drills, at least if it is stored as chemical energy. This is because drilling through terrain requires that some of the chemical bonds that hold the terrain together be broken, and so if the energy of chemical bonds is used to provide that power, then a given volume of chemical energy storage can only advance some fixed ratio of its length into the terrain, where the ratio is determined by the efficiency in taking bond energies of one sort to break bonds of a different sort. Based on these considerations, it appears unlikely that a completely self-contained subsurface vehicle could advance more than perhaps a hundred times its own length. Unless nuclear power sources are considered (and they have been), this requires some sort of tether to the surface to provide a nearly unlimited source of energy. Another problem for subsurface vehicles is that rock tends to expand when it is pulverized (in a process called *comminution*). Nonporous rock typically expands in volume by a few tens of percent when excavated, which means that fully self-contained subsurface vehicles have a severe *conservation of volume* problem. In principle the rock can be compressed back into its original volume, but this generally requires pressures much greater than the compressive strength of the original rock. The energy required to do this is much larger than the energy required to excavate the rock in the first place, and would become the dominant use of energy in an already energy-intensive effort.

As a result, it is generally assumed that any subsurface vehicle must keep some access tunnel open to the surface so that the excess volume of cuttings can be transported out. If this tunnel is available, then it seems that a means for getting power from the surface is also available, so that self-contained nuclear power is not needed. Subsurface vehicles with diameters as small as one or a few centimeters have been proposed that could potentially reach great depths within the mass and power constraints of feasible planetary robotic exploration missions.

55.5 Conclusions and Further Reading

Space robotics as a field is still in its infancy. The speed-of-light delays inherent in remote space operations makes problematic the master–slave teleoperation approach that has been very useful in the undersea and nuclear industries. Space robotics lacks the highly repetitive operations in a tightly structured environment that characterize industrial robotics. Hardware handled by space robots is very delicate and expensive. All three of these considerations have led to the fact that relatively few space robots have been flown, they have been very slow in operation, and only a small variety of tasks have been attempted. Nonetheless, the potential rewards of space robotics are great – exploring the solar system, creating vast space telescopes that may unlock the secrets of the universe, and enabling any viable space industries all seem to require major use of space robots. The scale of the solar system is not so great (a few light-hours) that human intelligence cannot always supplement even the most remote space robot that becomes confused or stuck. Indeed, for the Moon (with only a few seconds of time delay) it seems that hazard avoidance and reliable closure of force-feedback loops is all that is required to make a highly useful robotic system. For Mars (with tens of minutes of time delay), along with hazard avoidance and force loop closure, it seems that robust anomaly detection (with modest reflexive *safing* procedures) and perhaps scientific-novelty detection are probably all that is needed. High levels of autonomy are enhancing but not enabling for work in the inner solar system, and become more and more desirable for robots that are sent farther into the outer solar system.

For further reading, the following materials are suggested [55.109–114].

Video-References

VIDEO 330 DLR ROTEX: The first remotely controlled space robot
available from http://handbookofrobotics.org/view-chapter/55/videodetails/330

VIDEO 331 DLR predictive simulation compensating 6 seconds round-trip delay
available from http://handbookofrobotics.org/view-chapter/55/videodetails/331

Avidbots Ex1016
Page 1479 of 2154

VIDEO 332 DLR GETEX manipulation experiments on ETSVII
available from http://handbookofrobotics.org/view-chapter/55/videodetails/332
VIDEO 333 DLR ROKVISS animation
available from http://handbookofrobotics.org/view-chapter/55/videodetails/333
VIDEO 334 DLR ROKVISS camera images pulling spring
available from http://handbookofrobotics.org/view-chapter/55/videodetails/334
VIDEO 336 DLR ROKVISS disassembly
available from http://handbookofrobotics.org/view-chapter/55/videodetails/336
VIDEO 337 DLR telepresence demo remove cover
available from http://handbookofrobotics.org/view-chapter/55/videodetails/337
VIDEO 338 DLR telepresence demo with time delay
available from http://handbookofrobotics.org/view-chapter/55/videodetails/338
VIDEO 339 DLR DEOS demonstration mission simulation
available from http://handbookofrobotics.org/view-chapter/55/videodetails/339

References

55.1 D.L. Akin, M.L. Minsky, E.D. Thiel, C.R. Kurtzman: *Space Applications of Automation, Robotics and Machine Intelligence Systems (ARAMIS). Phase II* (NASA, Washington 1983)

55.2 C.G. Wagner-Bartak, J.A. Middleton, J.A. Hunter: Shuttle remote manipulator system hardware test facility, 11th Space Simulat. Conf. (1980) pp. 79–94, NASA CP-2150

55.3 S. Greaves, K. Boyle, N. Doshewnek: Orbiter boom sensor system and shuttle return to flight: Operations analyses, AIAA Guidance Navigation Contr. Conf. Exhibit. San Francisco (2005)

55.4 C. Crane, J. Duffy, T. Carnahan: A kinematic analysis of the space station remote manipulator system (SSRMS), J. Robotic Syst. **8**, 637–658 (1991)

55.5 M.F. Stieber, C.P. Trudel, D.G. Hunter: Robotic systems for the International Space Station, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 3068–3073

55.6 D. Bassett, A. Abramovici: Special purpose dexterous manipulator (SPDM) requirements verification, Proc. 5th Int. Symp. Artif. Intell. Robotics Autom. Space (1999) pp. 43–48, ESA SP-440

55.7 R. Boumans, C. Heemskerk: The European robotic arm for the International Space Station, Robotics Auton. Syst. **23**(1), 17–27 (1998)

55.8 P. Laryssa, E. Lindsay, O. Layi, O. Marius, K. Nara, L. Aris, T. Ed: International Space Station robotics: A comparative study of ERA, JEMRMS and MSS, Proc. 7th ESA Workshop Adv. Space Technol. Robotics Autom. (ASTRA) (2002)

55.9 T. Matsueda, K. Kuraoka, K. Goma, T. Sumi, R. Okamura: JEMRMS system design and development status, Proc. IEEE Natl. Telesyst. Conf. (NTC) (1991) pp. 391–395

55.10 S. Doi, Y. Wakabayashi, T. Matsuda, N. Satoh: JEM remote manipulator system, J. Aeronaut. Space Sci. Jpn. **50**(576), 7–14 (2002)

55.11 H. Morimoto, N. Satoh, Y. Wakabayashi, M. Hayashi, Y. Aiko: Performance of Japanese robotic arms of the International Space Station, 15th IFAC World Congr. (2002)

55.12 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: Sensor-based space robotics – ROTEX and its telerobotic features, IEEE Trans. Robotics Autom. **9**(5), 649–663 (1993)

55.13 M. Oda, K. Kibe, F. Yamagata: ETS-VII – Space robot in-orbit experiment satellite, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 739–744

55.14 K. Yoshida: Engineering test satellite VII flight experiments for space robot dynamics and control: Theories on laboratory test beds ten years ago, now in orbit, Int. J. Robotics Res. **22**(5), 321–335 (2003)

55.15 K. Landzettel, B. Brunner, G. Hirzinger, R. Lampariello, G. Schreiber, B.-M. Steinmetz: A unified ground control and programming methodology for space robotics applications – demonstrations on ETS-VII, Proc. Int. Symp. Robotics (ISR) (2000) pp. 422–427

55.16 J.C. Parrish, D.L. Akin, G.G. Gefke: The ranger telerobotic shuttle experiment: Implications for operational EVA/robotic cooperation, Proc. SAE Int. Conf. Environ. Syst. (2000)

55.17 J. Shoemaker, M. Wright: Orbital express space operations architecture program, Proc. SPIE **5088**, 1–9 (2003)

55.18 C. Preusche, D. Reintsema, K. Landzettel, G. Hirzinger: Robotics component verification on ISS ROKVISS – Preliminary results for telepresence, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4595–4601

55.19 K. Landzettel, A. Albu-Schaffer, C. Preusche, D. Reintsema, B. Rebele, G. Hirzinger: Robotic on-orbit servicing – DLR's experience and perspective, Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS) (2006) pp. 4587–4594

55.20 J.-H. Ryu, J. Artigas, C. Preusche: A passive bilateral control scheme for a teleoperator with time-varying communication delay, Mechatronics **20**(7), 812–823 (2010)

55.21 J. Artigas, J.-H. Ryu, C. Preusche, G. Hirzinger: Network representation and passivity of delayed teleoperation systems, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 177–183

55.22 E. Stoll, U. Walter, J. Artigas, C. Preusche, P. Kremer, G. Hirzinger, J. Letschnik, H. Pongrac: Ground

verification of the feasibility of telepresent on-orbit servicing, J. Field Robotics **26**(3), 287–307 (2009)
55.23 C. Ott, J. Artigas, C. Preusche, G. Hirzinger: Subspace-oriented energy distribution for the time domain passivity approach, Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS'11) (2011) pp. 665–671
55.24 A.P. Vinogradov: *Lunokhod 1 Mobile Lunar Laboratory* (JPRS, Washington 1971), JPRS identification number 54525
55.25 A. Mishkin: *Sojourner: An Insider's View of the Mars Pathfinder Mission* (Berkley Books, New York 2004)
55.26 B. Wilcox, T. Nguyen: Sojourner on mars and lessons learned for future planetary rovers, 28th Int. Conf. Environ. Syst. (1998)
55.27 M. Maimone, A. Johnson, Y. Cheng, R. Willson, L. Matthies: Autonomous navigation results from the Mars exploration rover (MER) mission, Proc. 9th Int. Symp. Exp. Robotics (ISER) (2004)
55.28 M. Maimone, Y. Cheng, L. Matthies: Two years of visual odometry on the Mars exploration rovers, J. Field Robotics **24**(3), 169–186 (2007)
55.29 M.G. Bekker: *Off-The-Road Locomotion* (Univ. Michigan Press, Ann Arbor 1960)
55.30 M.G. Bekker: *Introduction to Terrain-Vehicle Systems* (Univ. Michigan Press, Ann Arbor 1969)
55.31 R.A. Lewis, A.K. Bejczy: Planning considerations for a roving robot with arm, Int. Jt. Conf. Artif. Intell. (IJCAI) (1973) pp. 308–316
55.32 A. Thompson: The navigation system of the JPL Robot, Int. Jt. Conf. Artif. Intell. (IJCAI) (1977) pp. 749–757
55.33 Y. Yakimovsky, R.T. Cunningham: A system for extracting three-dimensional measurements from a stereo pair of TV cameras, Comp. Graph. Image Process. **7**, 195–210 (1978)
55.34 D. Dooling: *Planetary rovers might roam better with an elastic loop mobility system, NASA Science News, http://science.nasa.gov/newhome/ headlines/msad28apr98_1b.htm* 1989)
55.35 B.H. Wilcox: *Computer-Aided Remote Driving*, NASA Tech Briefs 18; 3 (NASA, Washington 1994)
55.36 E. Krotkov, J. Bares, T. Kanade, T. Mitchell, R. Simmons, W. Whittaker: Ambler: a six-legged planetary rover, 5th Int. Conf. Adv. Robotics, Robots Unstructured Environ. (ICAR), Vol. 1 (1991) pp. 717–722
55.37 J. Bares, W. Whittaker: Walking robot with a circulating gait, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (1990) pp. 809–816
55.38 B. Wilcox, D. Gennery: A Mars rover for the 1990s, J. Br. Interplanet. Soc. **40**, 484–488 (1987)
55.39 B. Wilcox, L. Matthies, D. Gennery: Robotic vehicles for planetary exploration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992)
55.40 D. Gennery, T. Litwin, B. Wilcox, B. Bon: Sensing and perception research for space telerobotics at JPL, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 311–317
55.41 J. Balaram, S. Hayati: A supervisory telerobotics testbed for unstructured environments, J. Robotics Syst. **9**(2), 261–280 (1992)
55.42 G. Giralt, L. Boissier: The French planetary rover VAP: concept and current developments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 1391–1398, LAAS Report No. 92227
55.43 R. Chatila, S. Lacroiz, G. Giralt: A case study in machine intelligence: Adaptive autonomous space rovers, Int. Conf. Field Serv. Robotics (FSR) (1997) pp. 101–108, LAAS Report No. 97463
55.44 A. Castano, A. Fukunaga, J. Biesiadecki, L. Neakrase, P. Whelley, R. Greeley, M. Lemmon, R. Castano, S. Chien: Autonomous detection of dust devils and clouds on Mars, Proc. IEEE Int. Conf. Image Process. (2006) pp. 2765–2768
55.45 L. Matthies: Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation, Int. J. Comput. Vis. **8**(1), 71–91 (1992)
55.46 K. Yoshida, D.N. Nenchev, M. Uchiyama: Moving base robotics and reaction management control, 7th Int. Symp. Robotics Res., ed. by G. Giralt, G. Hirzinger (Springer, New York 1996) pp. 101–109
55.47 J. Russakow, S.M. Rock, O. Khatib: An operational space formulation for a free-flying multi-arm space robot, 4th Int. Symp. Exp. Robotics (1995) pp. 448–457
55.48 Y. Xu, T. Kanade (Eds.): *Space Robotics: Dynamics and Control* (Kluwer, Boston 1993)
55.49 Y. Umetani, K. Yoshida: Continuous path control of space manipulators mounted on OMV, Acta Astronaut. **15**(12), 981–986 (1987), presented at the 37th IAF Conf, Oct. 1986
55.50 Y. Umetani, K. Yoshida: Resolved motion rate control of space manipulators with generalized Jacobian matrix, IEEE Trans. Robotics Autom. **5**(3), 303–314 (1989)
55.51 K. Yoshida: Impact dynamics representation and control with extended inversed inertia tensor for space manipulators, 6th Int. Symp. Robotics Res. (1994) pp. 453–463
55.52 K. Yoshida: Experimental study on the dynamics and control of a space robot with the experimental free-floating robot satellite (EFFORTS) simulators, Adv. Robotics **9**(6), 583–602 (1995)
55.53 Y. Nakamura, R. Mukherjee: Nonholonomic path planning of space robots via a bidirectional approach, IEEE Trans. Robotics Autom. **7**(4), 500–514 (1991)
55.54 S. Dubowsky, M. Torres: Minimizing attitude control fuel in space manipulator systems, Proc. Int. Symp. Artif. Intell. Robotics Autom. (i-SAIRAS) (1990) pp. 259–262
55.55 S. Dubowsky, M. Torres: Path planning for space manipulators to minimize spacecraft attitude disturbances, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1991) pp. 2522–2528
55.56 K. Yoshida: Practical coordination control between satellite attitude and manipulator reaction dynamics based on computed momentum concept, Proc. 1994 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1578–1585

Avidbots Ex1016
Page 1481 of 2154

55.57 M. Oda: Coordinated control of spacecraft attitude and its manipulator, Proc. 1996 IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 732–738

55.58 Z. Vafa, S. Dubowsky: On the dynamics of manipulators in space using the virtual manipulator approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 579–585

55.59 E. Papadopoulos, S. Dubowsky: Dynamic singularities in the control of free-floating space manipulators, ASME J. Dyn. Syst. Meas. Control **115**(1), 44–52 (1993)

55.60 Y. Umetani, K. Yoshida: Workspace and manipulability analysis of space manipulator, Trans. Soc. Instrum. Contr. Eng. **E-1**(1), 116–123 (2001)

55.61 D.N. Nenchev, Y. Umetani, K. Yoshida: Analysis of a redundant free-flying spacecraft/manipulator system, IEEE Trans. Robotics Autom. **8**(1), 1–6 (1992)

55.62 D.N. Nenchev, K. Yoshida, P. Vichitkulsawat, M. Uchiyama: Reaction null-space control of flexible structure mounted manipulator systems, IEEE Trans. Robotics Autom. **15**(6), 1011–1023 (1999)

55.63 V.H. Schulz: Reduced SQP Methods for Large Scale Optimal Control Problems in DAE with Application to Path Planning Problems for Satellite Mounted Robots, Ph.D. Thesis (Univ. Heidelberg, Heidelberg 1995)

55.64 E. Papadopoulos, S. Moosavian: Dynamics and control of multi-arm space robots during chase and capture operations, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1554–1561

55.65 K. Yoshida, D. Dimitrov, H. Nakanishi: On the capture of tumbling satellite by a space robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

55.66 C.G. Henshaw: A unification of artificial potential function guidance and optimal trajectory planning, 15th Space Flight Mech. Meet. (2005)

55.67 F. Aghili: A prediction and motion-planning scheme for visually guided robotic capturing of free-floating tumbling objects with uncertain dynamics, IEEE Trans. Robotics **28**(3), 634–649 (2012)

55.68 R. Lampariello, G. Hirzinger: Generating feasible trajectories for autonomous on-orbit grasping of spinning debris in a useful time, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013)

55.69 T. Binder, L. Blank, R. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schloeder, O. von Stryk: Introduction to model based optimization of chemical processes on moving horizons. In: *Online Optimization of Large Scale Systems*, ed. by M. Grotschel, S.O. Krumke, J. Rambau (Springer, Berlin, Heidelberg 2001) pp. 295–339

55.70 S. Jacobsen, C. Lee, C. Zhu, S. Dubowsky: Planning of safe kinematic trajectories for free-flying robots approaching an uncontrolled spinning satellite, ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. (DETC) (2002)

55.71 S.K. Agrawal, K. Pathak, J. Franch, R. Lampariello, G. Hirzinger: A differentially flat open-chain space robot with arbitrarily oriented joint axes and two momentum wheels at the base, IEEE Trans. Autom. Contr. **54**(9), 2185–2191 (2009)

55.72 D.N. Nenchev, K. Yoshida, M. Uchiyama: Reaction null-space based control of flexible structure mounted manipulator systems, IEEE 35th Conf. Decis. Control (1996) pp. 4118–4123

55.73 W.J. Book, S.H. Lee: Vibration control of a large flexible manipulator by a small robotic arm, Proc. Am. Control Conf. (1989)

55.74 M.A. Torres: Modelling, Path-Planning and Control of Space Manipulators: The Coupling Map Concept, Ph.D. Thesis (MIT, Cambridge 1993)

55.75 S. Abiko, K. Yoshida: An effective control strategy of japanese experimental module remote manipulator system (JEMRMS) using coupled and uncoupled dynamics, Proc. 7th Int. Symp. Artif. Intell. Robotics Autom. Space (2003), Paper AS18 (CD-ROM)

55.76 S. Abiko: Dynamics and Control for a Macro-Micro Manipulator System Mounted on the International Space Station, Ph.D. Thesis (Tohoku Univ., Tokyo 2005)

55.77 S. Abiko, K. Yoshida: On-line parameter identification of a payload handled by flexible based manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2930–2935

55.78 S. Abiko, K. Yoshida: An adaptive control of a space manipulator for vibration suppression, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

55.79 G. Gilardi, I. Shraf: Literature survey of contact dynamics modeling, Mech. Mach. Theory **37**, 1213–1239 (2002)

55.80 K. Yoshida, H. Nakanishi, H. Ueno, N. Inaba, T. Nishimaki, M. Oda: Dynamics, control, and impedance matching for robotic capture of a non-cooperative satellite, Adv. Robotics **18**(2), 175–198 (2004)

55.81 H. Nakanishi, K. Yoshida: Impedance control of free-flying space robot for orbital servicing, J. Robotics Mechatron. **18**(5), 608–617 (2006)

55.82 P.M. Pathak, A. Mukherjee, A. DasGupta: Impedance control of space robots using passive degrees of freedom in controller domain, J. Dyn. Syst. Meas. Control **127**, 564–578 (2006)

55.83 S. Abiko, R. Lampariello, G. Hirzinger: Impedance control for a free-floating robot in the grasping of a tumbling target with parameter uncertainty, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

55.84 K. Iagnemma, H. Shibly, S. Dubowsky: On-line traction parameter estimation for planetary rovers, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

55.85 K. Yoshida, H. Hamano: Motion dynamics of a rover with slip-based traction model, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

55.86 A. Jain, J. Guineau, C. Lim, W. Lincoln, M. Pomerantz, G. Sohl, R. Steele: Roams: Planetary surface rover simulation environment, Proc. 7th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS) (2003)

55.87 K. Yoshida, T. Watanabe, N. Mizuno, G. Ishigami: Terramechanics-based analysis and traction con-

Avidbots Ex1016
Page 1482 of 2154

trol of a lunar/planetary rover, Proc. Int. Conf. Field Serv. Robotics (FSR) (2003)
55.88 K. Iagnemma, S. Dubowsky: *Mobile Robots in Rough Terrain: Estimation, Motion Planning, and Control With Application to Planetary Rovers*, Springer Tracts in Advanced Robotics, Vol. 12 (Springer, Berlin, Heidelberg 2004)
55.89 K. Yoshida, G. Ishigami: Steering characteristics of a rigid wheel for exploration on loose soil, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004)
55.90 H. Shibly, K. Iagnemma, S. Dubowsky: An equivalent soil mechanics formulation for rigid wheels in deformable terrain, with application to planetary exploration rovers, J. Terramech. **42**(1), 1–13 (2005)
55.91 R. Bauer, W. Leung, T. Barfoot: Experimental and simulation results of wheel-soil interaction for planetary rovers, Proc. 2005 IEEE Int. Conf. Intell. Robots Syst. (IROS) (2005)
55.92 A. Ellery, N. Patel, R. Bertrand, J. Dalcomo: Exomars rover chassis analysis and design, Proc. 8th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS) (2005)
55.93 A. Gibbesch, B. Schäfer: Multibody system modelling and simulation of planetary rover mobility on soft terrain, Proc. 8th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS) (2005)
55.94 G. Ishigami, K. Yoshida: Steering characteristics of an exploration rover on loose soil based on all-wheel dynamics model, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)
55.95 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil, J. Field Robotics **24**(3), 233–250 (2007)
55.96 V.K. Tiwari, K.P. Pandey, P.K. Pranav: A review on traction prediction equations, J. Terramech. **47**(3), 191–199 (2010)
55.97 L. Ding, H. Gao, Z. Deng, K. Nagatani, K. Yoshida: Experimental study and analysis on driving wheels' performance for planetary exploration rovers moving in deformable soil, J. Terramech. **48**(1), 27–45 (2011)
55.98 G. Meirion-Griffith, M. Spenko: A modified pressure-sinkage model for small, rigid wheels on deformable terrains, J. Terramech. **48**(2), 149–155 (2011)
55.99 J.Y. Wong: *Theory of Ground Vehicles* (Wiley, New York 1978)
55.100 J.Y. Wong, A.R. Reece: Prediction of rigid wheel preformance based on the analysis of soil-wheel stresses. Part I, Preformance of driven rigid wheels, J. Terramech. **4**, 81–98 (1967)
55.101 Z. Janosi, B. Hanamoto: The analytical determination of drawbar pull as a function of slip for tracked vehicle in deformable soils, Proc 1st Int. Conf. Terrain-Veh. Syst. (1961)
55.102 E. Hegedus: *A Simplified Method for the Determination of Bulldozing Resistance*, Land Locomotion Laboratory, Rep. No. 61U.S. (Army Tank Automotive Command, Warren 1960)
55.103 I. Rekleitis, E. Martin, G. Rouleau, R. L'Archeveque, K. Parsa, E. Dupuis: Autonomous capture of a tumbling satellite, J. Field Robotics **24**(4), 275–296 (2007)
55.104 J. Blamont: Balloons on other planets, Adv. Space Res. **1**, 63–69 (1981)
55.105 J.A. Cutts, K.T. Nock, J.A. Jones, G. Rodriguez, J. Balaram, G.E. Powell, S.P. Synott: Aerovehicles for planetary exploration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995)
55.106 M.K. Heun, J.A. Jones, J.L. Hall: Gondola design for Venus deep-atmosphere aerobot operations, AIAA 36th Aerosp. Sci. Meeting Exhibit. ASME Wind Energy Symp. (1998)
55.107 S. Miller, J. Essmiller, D. Beaty: Mars deep drill – A mission concept for the next decade, Space Conf. Exhibit. (2004) pp. 2004–6048
55.108 S. Mukherjee, P. Bartlett, B. Glass, J. Guerrero, S. Stanley: Technologies for exploring the martian subsurface, Proc. IEEE Aerosp. Conf. (2006), Paper No. 1349
55.109 C.F. Ruoff, S.B. Skaar (Eds.): *Teleoperation and Robotics in Space*, Progress in Astronautics and Aeronautics Series (AIAA, Notre Dame 1994)
55.110 J. Field Robot. Special Issue on Space Robotics, Part I–Part III, **24**(3–5), 167–434 (2007)
55.111 G.F. Bekey (Ed.): *International Assessment of Research and Development in Robotics* (WTEC, Baltimore 2006), http://www.wtec.org/robotics/
55.112 NSRC (Ed.): *Assessment of Options for Extending the Life of the Hubble Space Telescope*, National Research Council: Final Rep. (National Academies Press, Washington 2005)
55.113 A.M. Howard, E.W. Tunstel: *Intelligence for Space Robotics* (TSI, San Antonio 2006)
55.114 T.B. Sheridan: Space teleoperation through time delay: Review and prognosis, IEEE Trans. Robotics Autom. **9**(5), 592–606 (1993)

Avidbots Ex1016
Page 1483 of 2154

Multimedia Contents

56. Robotics in Agriculture and Forestry

Marcel Bergerman, John Billingsley, John Reid, Eldert van Henten

Robotics for agriculture and forestry (A&F) represents the ultimate application of one of our society's latest and most advanced innovations to its most ancient and important industries. Over the course of history, mechanization and automation increased crop output several orders of magnitude, enabling a geometric growth in population and an increase in quality of life across the globe. Rapid population growth and rising incomes in developing countries, however, require ever larger amounts of A&F output. This chapter addresses robotics for A&F in the form of case studies where robotics is being successfully applied to solve well-identified problems. With respect to plant crops, the focus is on the in-field or in-farm tasks necessary to guarantee a quality crop and, generally speaking, end at harvest time. In the livestock domain, the focus is on breeding and nurturing, exploiting, harvesting, and slaughtering and processing. The chapter is organized in four main sections. The first one explains the scope, in particular, what aspects of robotics for A&F are dealt with in the chapter. The second one discusses the challenges and opportunities associated with the application of robotics to A&F. The third section is the core of the chapter, presenting twenty case studies that showcase (mostly) mature applications of robotics in various agricultural and forestry domains. The case studies are not meant to be comprehensive but instead to give the reader a general overview of how robotics has been applied to A&F in the last 10 years. The fourth section concludes the chapter with a discussion on specific improvements to current technology and paths to commercialization.

56.1 **Section Scope** 1464
56.2 **Challenges and Opportunities** 1465
56.3 **Case Studies** 1467
56.3.1 Optimized Coverage for Arable Farming 1467
56.3.2 Weed Control 1468
56.3.3 High Precision Seeding 1470
56.3.4 Crop Yield Estimation 1471
56.3.5 Precision Irrigation 1471
56.3.6 Tree Fruit Production 1472
56.3.7 Vehicle Formation Control 1473
56.3.8 Date Palm Tree Spraying 1474
56.3.9 Plant Probing 1475
56.3.10 Cucumber Harvesting 1476
56.3.11 Cucumber Leaf Removal 1477
56.3.12 Rose Harvesting 1477
56.3.13 Strawberry Harvesting 1477
56.3.14 Pot Handling in Nurseries and Greenhouses 1478
56.3.15 Precision Forestry 1479
56.3.16 Semi-Automation of Forwarder Crane 1480
56.3.17 Livestock Breeding and Nurturing 1482
56.3.18 Livestock Exploitation 1483
56.3.19 Livestock Harvesting, Slaughtering, and Processing 1484
56.3.20 Aerial-Based Precision Agriculture 1485
56.4 **Conclusion** 1487
Video-References 1488
References 1489

Avidbots Ex1016
Page 1484 of 2154

Robotics for agriculture and forestry represents the ultimate application of one of our society's latest and most advanced innovations to its most ancient industries. Since the dawn of civilization, agriculture and forestry (A&F for short) remain chief among humankind's most important economic activities, providing the food, feed, fiber, and fuel necessary for our survival. (In this chapter, agriculture is understood as in the Merriam-Webster definition:

> *the science, art, or practice of cultivating the soil, producing crops, and raising livestock and in varying degrees the preparation and marketing of the resulting products.*

Therefore, the term *crop* may be used here to denote any product of an agricultural or forestry process, including grains, cereals, fruit, vegetables, nuts, trees, beef, wool, etc. Whenever necessary, we will differentiate plant from animal products appropriately.)

Over the course of history, mechanization and automation – from the manual ploughs of yore to the modern combines of today – increased crop output several orders of magnitude. This, in turn, enabled a geometric growth in population and a corresponding increase in quality of life across the globe. Rapid population growth and rising incomes in developing countries, however, require ever larger amounts of A&F output. Scientists predict that agricultural production must double to meet the demands of nine billion people in 2050 [56.1–3]. Clearly, this cannot be achieved by simply doubling the inputs (land, water, seeds, labor, etc.) because of constrained resources and environmental concerns. Therefore, the efficiency of the A&F system must increase in a sustainable and consistent manner.

According to *Reid* [56.4], global agricultural total factor productivity (TFP), or the output per unit of total resources used in production, must increase from the current 1.4 to 1.75 to double agricultural output by 2050. This requires significant scientific, technologic, and management advances in all of the factors that impact TFP – seeds, soil, water, fertilizers, herbicides, insecticides, crop architecture, cultural practices, automation, labor, public policy, etc. While robotics is but one of these factors, it is one with the potential to effect A&F in a broad, systemic way, and contribute significantly to meeting our future needs.

In this chapter we address the field of agricultural and forestry robotics *from the point of view of the applications it enables*, rather than from the point of view of the elementary technologies it comprises. Furthermore, we chose to present a limited number of case studies where robotics is being successfully applied to solve well-identified problems, rather than a comprehensive survey of all work reported in the literature. We believe the former is more meaningful as it showcases top-down, problem-oriented solutions (*market pull*) rather than bottom-up, technology-led ones (*technology push*). We focus on examples from the last five to ten years, as they represent work that leverages the most recent advances in sensors and computing.

We address applications of agricultural and forestry robotics in the form of twenty case studies where robotics is being successfully applied to solve well-identified problems. With respect to plant crops, the focus is on the in-field or in-farm tasks necessary to guarantee a quality crop and, generally speaking, end at harvest time, before the crop is transported to a packing plant or warehouse. In the livestock domain, the focus is on breeding and nurturing, exploiting, harvesting, and slaughtering and processing.

56.1 Section Scope

Before presenting the case studies that showcase recent advances in robotics applied to A&F, we must define the scope of the chapter, both in terms of crop production processes and robotic technologies.

A typical crop production cycle includes several processes, among them field preparation, seeding/breeding, transplanting, planting, growing, maintenance (including attaching plants to support structures, disbudding, removing leaves, pruning tree limbs and shoots, thinning blossoms and fruit, nurturing animals, etc.), exploiting/harvesting/slaughtering, sorting, and packing. In all of them, internal transport of people, machines, and produce play a role. Single-harvest crops such as lettuce must be replaced once harvested; multiple-harvest crops such as apples, tomatoes, and roses last a year or even several years before they need to replanted. Depending on the crop, machinery with varying levels of automation exists for some or all of these processes. In grain and cereal production, for example, farmers have access to commercial machines for tilling, seeding, transplanting, spraying, irrigating, and harvesting. In fresh fruits and vegetables, on the other hand, mechanization and automation are more prevalent at the early and late stages of the production cycle [56.5], with crop maintenance and harvest remaining for the most part manual tasks.

With respect to plant crops, this chapter's scope is limited to the in-field or in-farm tasks necessary

to guarantee a quality crop and, generally speaking, end at harvest time, before the crop is transported to a packing plant or warehouse. Of course, there are many opportunities for robotics in those latter environments, e.g., automatic sorting and grading. We exclude them from our presentation as they are currently much more the focus of automation efforts than of robotics per se. In future editions of the Handbook, we will revisit this separation as robotics advances to post-harvest tasks.

In the livestock domain, much of farming concerns the nurturing, exploitation, harvesting, and slaughtering of animals. Slaughtering extends beyond the killing of the animal to its division into marketable portions, and subsequent packing and marketing. The essential processes considered in this section pertain to land animals and birds and include:

- Breeding and nurturing: Livestock can range from poultry to cattle in feedlots, including pigs and other animals. Their care involves environment and behavior monitoring, plus fodder distribution.
- Exploiting: In many cases, products are obtained from the live animals. Sheep are shorn for wool, cows are milked, chickens lay eggs, and bees produce honey.
- Harvesting: Mustering and collecting *free-range* cattle, feral pigs, and other unconstrained animals.
- Slaughtering and processing: Cattle, poultry, pigs, other animals are killed subject to strict regulations, then divided into marketable portions.

In a future edition of the Handbook fishing should be included, since many opportunities for robotics can be imagined there.

56.2 Challenges and Opportunities

Since the end of the industrial revolution, (arguably) the three most significant impacts robotics and automation have had on agriculture and forestry are:

1. Precision agriculture, or the use of sensors to precisely control when and where to apply inputs such as fertilizers and water
2. Auto-guidance on field crop machinery, which today can drive down a field with an accuracy unattainable by human drivers
3. Machines that harvest fruits and vegetables for processing (e.g., tomato paste and orange juice).

Academic and commercial researchers are now focusing on the next wave of *sensing*, *mobility*, and *manipulation* technologies that promise to increase A&F output and productivity.

Sensing entails measuring crop temperature, humidity, pH, wetness, image, range, and other physical attributes, and combining and analyzing the data for specific purposes. One example is a camera-based system that takes pictures of an apple orchard a few weeks before harvest and produces an accurate crop yield estimate that growers can use to plan and manage the harvest operation [56.6, 7]. The utility of sensing to A&F is that it enables decision-making at a level unattainable by human sensing alone, because the latter is either inherently inaccurate or slow or both.

Mobility relates to various levels of vehicle automation that enable driverless (or driver-assistive) field coverage; the most common example is a global positioning system/global navigation satellite system (GPS/GLS)-guided combine that harvests corn with minimal intersection between passes on the field, thus optimizing coverage and minimizing fuel consumption [56.8]. More recently, auto-guidance has started to migrate to orchard vehicles as well, albeit here other navigation sensors may be required because of poor satellite reception under thick canopies (VIDEO 26, VIDEO 91). In A&F, automated vehicles equipped with the appropriate implements enable (semi-) autonomous seeding, spraying, mowing, weed removal, harvesting, and animal feeding, among other operations (VIDEO 306, VIDEO 305). Mobility requires some level of sensor-based perception, provided by GPS/GNSS, inertial units, cameras, ladars, radars, etc. These sensors are not to be confused with those described in the previous paragraph, although there certainly are situations where a sensor can perform double duty of sensing for decision making and for navigation. In general, robotic mobility technology is currently less advanced than sensing.

Manipulation refers to the various operations performed directly on the crop, including pruning, thinning, harvesting, tree training, leaf probing, tree cutting, weed removal, etc. In general, this technology requires more sophisticated sensor-based perception than mobility, and in terms of field deployment is less advanced than either sensing or mobility.

Two domains that are particularly challenging for robotics research are orchard crops and crops in protected cultivation. (We use the term *orchard* liberally to include grape vineyards and orange groves, among other similar environments where crops are grown in well-defined rows.) These are highly valuable crops, potentially generating one to three orders of magnitude more revenue per acre than field crops. They are also

characterized by the need for intensive cultivation and skilled labor. Consider, for example, apple production in the US – no less than 50 to 60% of the variable costs to produce an apple is due to labor [56.9]. On top of that, labor is needed in bursts – in Washington state, in the US, up to seven times more workers are needed during the fall harvest season than during the winter pruning season. In many developed countries, labor availability for manual orchard operations is a challenge, putting significant pressure on growers to find innovative solutions to address their labor requirements. In the 2011 Tree Fruit Industry Perspective publication, industry leaders with the US Northwest Farm Credit Services say that:

> *[...] seasonal labor force utilized by the tree fruit industry for thinning and harvest operations will always be an issue of concern [...]. Labor shortages at various times throughout the growing season have occurred and will likely occur in the future. The perishable nature of the tree fruit mandates a strict timeliness in field operations, and the generally narrow harvest windows do not accommodate labor shortages at critical stages. [...] New technology [...] could reduce labor requirements substantially over the next five to 10 years. Specifically, such technology would include the use of platforms and mechanical-harvest methods in the orchards, and increased use of robotics [...].*

Partly in response to the high labor costs and partly to increase production efficiency, the tree fruit industry is moving in the direction of highly structured planting architectures. Whereas before an apple grower had two to four hundred trees per acre, she now has 1200 apple trees per acre in a *fruit wall* configuration that is much more conducive to automation (Fig. 56.1a). This has led to the development of autonomous vehicles and tractors that drive from garage to block and traverse tree rows for hours on end, including turning from row to row, without any human assistance [56.10]. When equipped with the appropriate implements, these robotic vehicles can mow, spray, and collect tree and crop data for inventory management; and when configured as platforms, they can carry workers pruning, thinning, training trees, and harvesting the top part of the trees, thus eliminating the inefficiencies and injuries related to ladders. In the future, manipulators mounted on such vehicles will be able to probe plants for phenotyping purposes, and automatically prune, thin, and harvest.



Fig. 56.1 **(a)** Modern apple orchard with trees arranged to form a *fruit wall*. **(b)** Vine tomato in a protected cultivation environment. These new architectures open the door to robotics-based production technologies that increase efficiency and reduce labor costs

A protected cultivation system is a powerful instrument to produce crops (Figure 56.1b). Greenhouses protect crops from unfavorable climate conditions and pests and offer the opportunity to modify the climate to create an environment that is optimal in terms of both crop quality and quantity. A protected cultivation is an intensive production method with high investment and operational costs, therefore permitting only the production of high value fruit and vegetable crops such as tomatoes, sweet peppers and cucumbers, flowers such as roses, chrysanthemums and gerberas and many types of potted plants. In the past decades, in western societies, this type of production has been confronted with increasing size of production facilities, increasing labor costs, reduced availability of sufficiently skilled labor, health problems of the employees due to heavy and repetitive tasks, and growing competition on the national and international markets. Automation and robotics are considered to be a way to address these issues. Additionally, growing concerns with food safety call for such technology. Last but not least, more and more a precision horticulture approach is adopted in which plants are treated on an individual basis, to improve quantity and quality of crop production whilst using resources as efficiently as possible. Given the current constraints on human labor, this has led to an even stronger call for automation and robotics.

Within the context of agricultural production, in terms of production area, protected cultivation is worldwide a relatively small business. Total area in use with protected cultivation worldwide is estimated at roughly 740 000 ha [56.11]. In terms of added value, however, protected cultivation plays a much more important role. In The Netherlands at a few percent of the area available for agricultural production, horticulture produces roughly 35% of the economic return of total agricultural production. In that case production is very capital and labor intensive. On a global scale, the potential of protected cultivation is being increasingly acknowledged. Yet, technology levels still vary widely, largely based on significant differences in local conditions in terms of market, economy, resource availability, etc.

Robotics research in protected cultivation has a history of some thirty years [56.12], focused mainly on

harvesting and chemical spraying operations. Robotic harvesting focused mainly on tomatoes, cucumbers, eggplants, sweet peppers, and strawberries, for which multiple examples of research prototypes have been developed [56.12] (VIDEO 304). Single examples are known of a harvesting robot for roses [56.13] and gerbera [56.14]. Spraying robots have been reviewed in detail in [56.15]. Only two robotic systems, other than for harvesting or spraying, were found in the literature: a leaf picking robot for cucumber [56.16] and a robot for bagging, thinning, and spraying in grapes [56.17]. In protected cultivation robots have not yet been developed for actions like pruning, disbudding and attaching of plants to support structures.

Challenges in livestock production are of a more practical nature. When cattle roam freely they can be difficult to find and muster. Spotting by helicopter is expensive so there is great potential for the use of unmanned aerial vehicles. The method of collecting feral pigs and native kangaroos is usually by shooting, supplying markets in Germany and Russia respectively. There could in theory be an opportunity for the application of robotics, but it is difficult to see how this could be compatible with safety. It has been suggested that cattle could be located by carrying a transmitter. Radio communication involves considerable distances, so the use of GPS/GNSS collars with transmitting systems would pose interesting power and battery problems.

56.3 Case Studies

In this section we present a variety of case studies where robotics technologies are being applied successfully to problems in the agriculture and forestry sectors. The presentation follows a somewhat loose categorization by application domain, i. e., field crops, orchard crops, protected cultivation, forestry, and livestock. The point here is to group examples that are domain-specific but allow for the discussion of those that straddle across domains. Within each domain, the case studies follow (again, loosely) the progression from sensing to mobility to manipulation.

56.3.1 Optimized Coverage for Arable Farming

Agricultural researchers and practitioners have long-desired to have the capability to follow well-defined traffic lanes with common-width equipment systems to minimize soil compaction effects on plant growth [56.18]. The advent of automation and control technology (e.g., GNSS, automatic steering) eliminated the need for complex processes to define the controlled-traffic lanes by making it possible for agricultural machine systems to follow precise paths spatially and temporally.

With the rapid adoption of automatic guidance systems, automated path planning has great potential to further optimize field operations. Field operations should be done in a manner that minimizes time and travel over field surfaces and is coordinated with specific field operations, machine characteristics, and topographical features of arable lands. To reach this goal, *Jin* and *Tang* [56.19] proposed an optimal coverage path planning (OCPP) algorithm where coverage is represented by a geometric model. To determine the full coverage pattern of a given field by using boustrophedon paths, it is necessary to know whether and how to decompose a field into sub-regions and how to determine the travel direction within each sub-region. The search mechanism is guided by a customized cost function resulting from the analysis of different headland turning types and implemented with a divide-and-conquer strategy. The complexity of the algorithm is $O(n^3 \log(n^3))$ for a field with n edges. In order to reduce the total turning cost, the number of turns needed to be minimized. Besides, turns with relatively high operational costs needed to be avoided. Fields of irregular shapes had inefficiencies related to headland turns when headlands were at an angle to the machine. Two-dimensional field examples with complexity ranging from a sim-

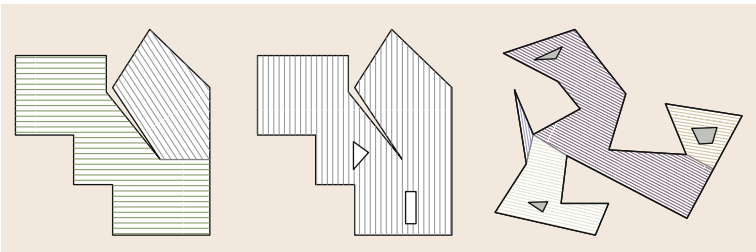


Fig. 56.2 Results obtained with the optimal coverage path planning algorithm for two-dimensional (2-D) terrains; the inner polygons indicate non-traversable obstacles

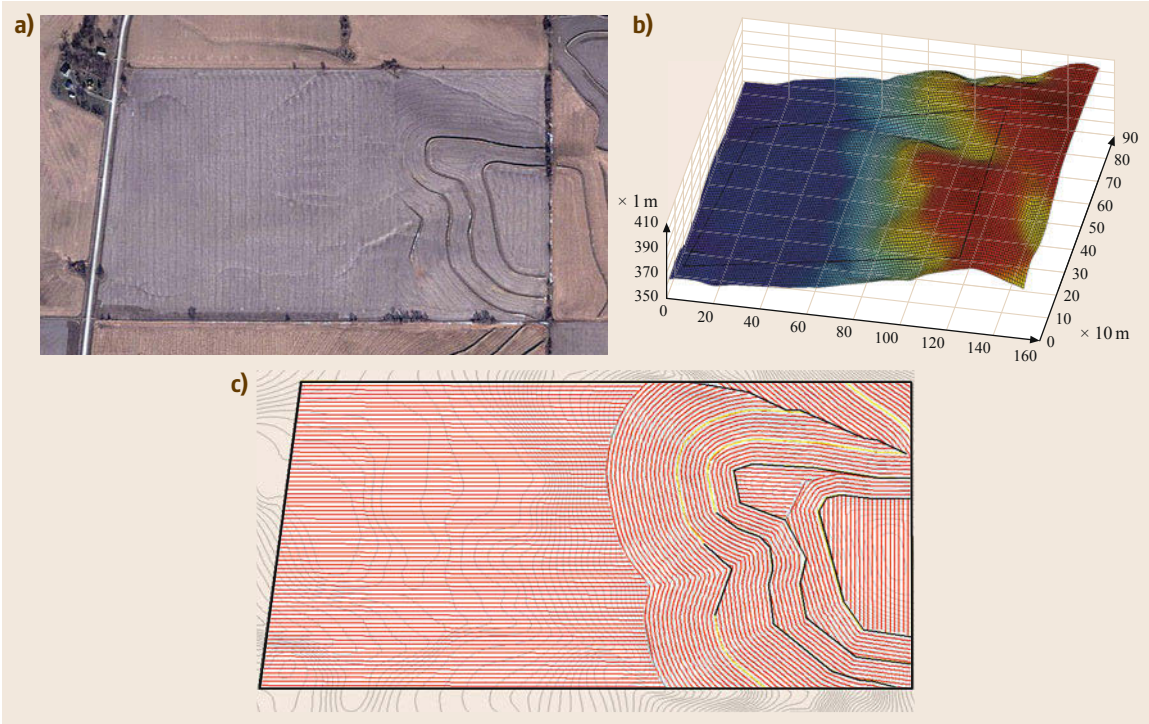


Fig.56.3a–c Results obtained with the optimal coverage path planning algorithm for a 3-D terrain with terraces and valleys. **(a)** Aerial image. **(b)** Topographic map. **(c)** Optimal coverage path

ple convex shape to an irregular polygonal shape that has multiple obstacles within its interior were tested with the OCPP algorithm (Fig. 56.2). The results show that, in the most extreme two-dimensional cases, OCPP saved up to 16% in the number of turns and 15% in headland turning cost. There were no cases where OCPP outputted solutions worse than those adopted by farmers.

When optimizing coverage path over three-dimensional (3-D) terrains, more factors need to be considered, including headland turning cost, soil erosion cost, and skipped area cost. Jin and Tang developed an analytical 3-D terrain model with B-splines; and analyzed different categories of coverage costs on 3-D terrains and developed methods to quantify soil erosion cost and curving path cost corresponding to a particular coverage solution. Similar to the 2-D coverage path optimization, terrain decomposition and classification methods are used to divide a field into sub-regions with similar field attributes and comparatively smooth boundaries. The divide-and-conquer strategy was also applied in the 3-D terrain coverage planning. The most appropriate path direction of each region was the one that achieved the minimum coverage cost. A *seed curve* searching algorithm was successfully developed and applied to several practical farm fields with various topographic features (Fig. 56.3). The 3-D planning algorithm has shown its superiority on 3-D terrain fields compared with the 2-D planner. On the tested fields, on average the 3-D version saved 10.3% on headland turning cost, 24.7% on soil erosion cost, 81.2% on skipped area cost, and 22.0% on the weighted sum of these costs, where their corresponding weights were 1, 1, and 0.5, respectively. In one of the regions, in particular, the 3-D planning algorithm generated a result with only 30.5% of the soil erosion caused by the 2-D planning algorithm. It was also observed that the skipped area resulting from the sharp turning curvature in 3-D planning is generally much smaller than the skipped area between paths when projecting 2-D planning results to a 3-D surface.

56.3.2 Weed Control

Weeds compete with the production crop for light, water, and nutrients and can have a detrimental impact on crop yield if uncontrolled [56.20]. For these reasons chemical and mechanical weed control have long received attention of agricultural engineers. Compared to current methods, robotics offers the opportunity to improve this important production task in two ways: greater precision when done mechanically, and reduced

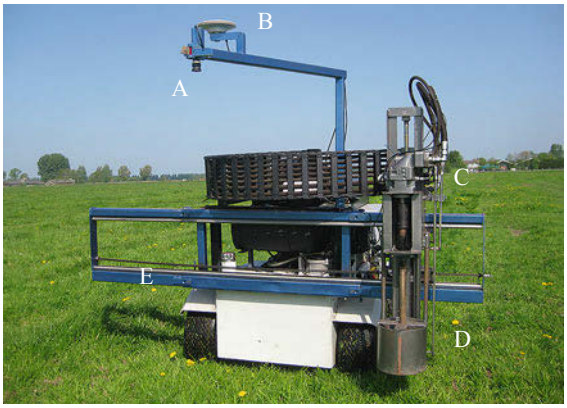


Fig. 56.4 A robot to detect and control broad-leaved dock in grassland. Weeds are detected with a texture-based algorithm using images from a color camera (A). Navigation is GPS-based (B). A diesel engine (C) provides hydraulic power for the weeding implement (D), which can be moved laterally along the rail (E)

emissions and environmental runoff when done chemically. This section describes three examples of the application of robotics to weed control; for an overview of the field see [56.20].

Broad-leaved dock is a troublesome weed in grassland. When not controlled it may reach a high density and considerably reduce grass yield. In response to a request of seventeen organic Dutch dairy farmers, a robot was constructed to detect and control broad-leaved dock in grassland [56.21] (Fig. 56.4) (VIDEO 310). It consists of a custom platform with four independently driven wheels powered by a 36 kW diesel engine. The wheels are fitted with golf cart tires that provide traction on grass with minimal impact on the sod. Reduction gears ensure high torque, allow centimeter-precision forward movement, and limit maximum speed to a safe 3 mph. Skid steering was deemed sufficiently precise under the circumstances. Traversal of a large, featureless pasture was achieved by using real-time kinematic GPS (RTK-GPS) to follow a pre-defined path consisting of parallel segments connected by half-circle turns at the ends. Because weeds and grass are both green, a texture-based image analysis method was used to detect the former [56.22, 23]. The weeder proposed by Riesenhuber was used [56.24]. The method consists of a rod weeder driven vertically into the ground to fragment the weed's taproot.

Volunteer potatoes are the leftovers of the previous year's harvest. After a mild winter these potatoes will sprout and constitute a serious weed. They not only compete with the production crop, but also potentially carry diseases. In The Netherlands, legislation requires that they be removed annually by July 1st. This labor-intensive task naturally called for automation. As a joint effort of industry, policy makers, and scientists, a project to detect and control volunteer potato plants was initiated [56.25–27]. A proof-of-principle machine for sugar beet fields has been built and tested (Fig. 56.5). Machine vision-based detection at 100 mm^2 precision was combined with a micro-sprayer with five needles and a working width of 0.2 m. The accuracy of the system was ± 14 mm in the longitudinal direction and ± 7.5 mm in the transversal direction. The main error source was the variability in micro-sprayer droplet velocity that caused longitudinal errors. Still, 77% of volunteer plants with a size larger than 1200 mm^2 were successfully controlled at velocities up to 0.8 $m\,s^{-1}$. Within the seed line, glyphosate was applied on weed potato plants, accompanied with up to 1.0% unwanted killed sugar beet plants.

The Intelligent Autonomous Weeder (Fig. 56.6) is a four-wheel steer, four-wheel drive platform to be used for autonomous weeding operations in arable farming [56.28–30]. The platform combines dual GPS-

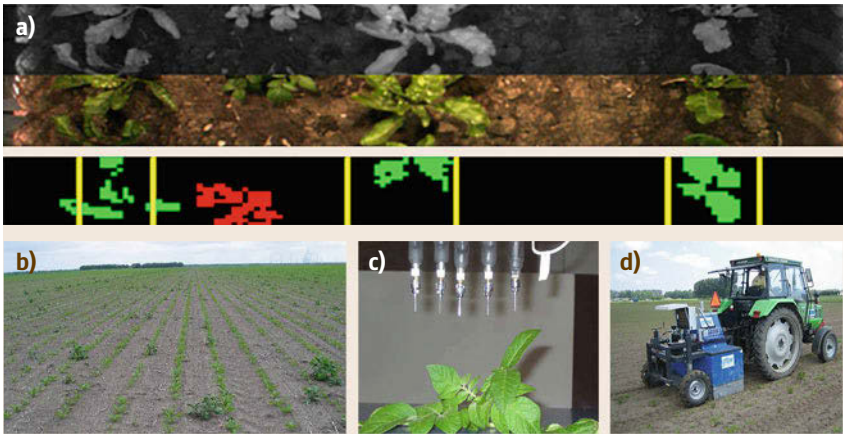


Fig.56.5a–d Precision removal of volunteer potatoes in sugar beets. **(a)** Computer vision-based detection of the potato weeds (*red*) among the sugar beet plants (*green*). **(b)** A sugar beet field infested with potato weeds. **(c)** A micro-spray system. **(d)** The removal device pulled by a tractor



Fig. 56.6 The Intelligent Autonomous Weeder operates in arable farming land

based navigation with computer vision-based row following. The four wheel steering construction offers supreme maneuverability, which is not only an advantage when precise operation within the crop is considered, but also offers the opportunity of very compact head land turns.

56.3.3 High Precision Seeding

The premise behind this work is that, if an autonomous agricultural machine could accurately follow a predefined path to carry out seeding, then the same machine could drive in the field throughout the entire growing season carrying out subsequent tasks – e.g., weeding, fertilizing, spraying, etc. – without the need for repeated crop location sensing (VIDEO 131). The primary challenge is to develop centimeter-level path tracking accuracy along straight runs by the mobile agricultural machines, and even greater accuracy (1–2 cm) in agricultural tool manipulation, e.g., controlling seeding tines for seed placement.

While there are semi-automated seeding systems currently on the market, they suffer from a number of problems with major implications for environmental sustainability, productivity, and economic return:

- They are passive, i. e., unable to take action to correct the seeding tines' path.
- The seeding implements are towed by a tractor which is GPS-guided, but with insufficient accuracy (usually 40 cm); moreover, accurate path following by a tractor does not guarantee accurate seed placement by the implement.
- The tractor does not sense deviation of the seeding implement's path, causing the crop to be laid out unpredictably.
- Even when the system senses implement deviation, it does not have the ability to correct the seeding implement's path, let alone the position of the seeding tines.
- Tractors used in current systems are massive, therefore must use fixed tracks to limit ground compaction (however still causing compaction of up to 20% of arable land).
- Insufficient seeding accuracy does not permit inter-row cropping in alternating seasons – a technique that utilizes remnant nutrients in inter-row spaces, and
- Present day systems do not have fully autonomous operational capability.

Katupitiya et al. [56.31] built and delivered a system that advanced the field in the following ways (Fig. 56.7):

- An active seeder equipped with sophisticated control systems that can take corrective action against path deviations while ensuring high accuracy seeding tine position control.
- High accuracy localization of the seeder and tines achieved via precise GPS, high-precision sensors, data fusion, and control software to locate/position the seeding tines. Control systems include those that micro-adjust the seeding tines with respect to the main seeder frame. This allows the seeding tine positions to follow a more accurate path than the seeder itself.
- The seeder is force-controlled and self-propelled, allowing for the size of the tractor to be greatly reduced. Smaller tractors also mean smaller wheels, so the tractor wheel width can be smaller than the inter-crop row width; hence the tractor can use the inter-row space as its wheel tracks without squashing the crop for subsequent operations.



Fig. 56.7 Precision seeding system for broad acre farming

These result in the complete elimination of ground compaction.

- The level of automation integrated into the tractor-seeder pair is such that the entire system is readily autonomous with no operator required.
- The seeder is modular, with either single or multiple units operating around the clock. A special purpose tandem, non-linear adaptive pursuit path tracking algorithm is used to control the steering of the tractor and the seeder wheels. The seeder has its wheels under force control based on the tension at the off-axle hitch point.

56.3.4 Crop Yield Estimation

One common desire of all fruit growers is knowledge of the crop yield. Accurate yield prediction helps growers improve fruit quality and reduce operating cost by making better decisions on intensity of fruit thinning and size of the harvest labor force. It benefits the packing industry as well, because managers can use estimation results to optimize packing and storage capacity.

Typically, yield estimation is performed based on historical data, weather conditions, and workers manually counting fruit in multiple sampling locations. Manually gathering samples is a time-consuming, labor-intensive, and inaccurate process, and the number of samples is usually far too small to capture the magnitude of the variation in yield across each block. Growers are searching for an automated and efficient alternative that can accurately capture spatial variation in yield.

To deal with this need, *Nuske* et al. [56.6, 7] developed a computer vision-based system to detect and count fruit. The system uses a camera rig for image acquisition, working at nighttime with controlled artificial lighting to reduce the variance of natural illumination. An autonomous vehicle is used as the support platform for automated data collection (Fig. 56.8). The system scans both sides of each row of trees or vines, detecting fruit captured within the image sequence, and then generates yield estimates.

The accuracy of the system was demonstrated by comparing its crop yield estimation with ground-truth recorded via careful and tedious manual measurements. Its end result is a yield map that closely resembles the true spatial distribution of yield, which growers can utilize to make critical production management decisions.

The yield estimation system was deployed in a number of vineyards, apple orchards, and strawberry ranches, and the results show that the system works well in a variety of crops and training structures (Figs. 56.9 and 56.10).

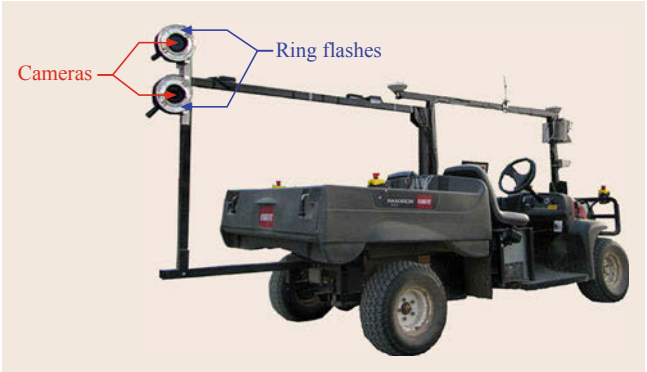


Fig. 56.8 Autonomous crop yield estimation system hardware, composed of concentric cameras and ring flashes for controlled illumination during nighttime operations

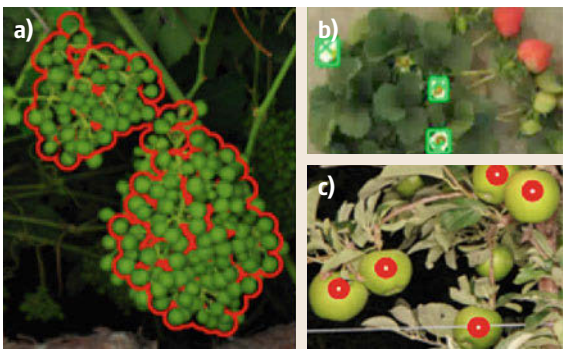


Fig.56.9a–c Output of the crop yield estimation system in vineyards **(a)**, apple orchards **(c)**, and strawberry ranches **(b)**

56.3.5 Precision Irrigation

Current irrigation practice within the agricultural community often dictates over-watering crops as opposed to under-watering. This results in wasted resource, increased leaching of fertilizers, and an increase in crop disease. There are crop models that determine adequate irrigation quantities, but they are seldom used as the constant modifying of irrigation parameters is tedious and difficult.

Kohanbash et al. [56.32] used wireless sensor networks (WSN's) to monitor environmental conditions in real time and adjust irrigation parameters on the fly. WSN's can do basic set-point control where irrigation is enabled every time the soil moisture goes below a predefined threshold. More advanced control methods are also possible by integrating crop water use models into the system. WSN systems communicate with a central base station that can be connected to the Internet, allowing remote access for viewing crop condition and modifying irrigation settings. Having a central base sta-

Avidbots Ex1016
Page 1492 of 2154

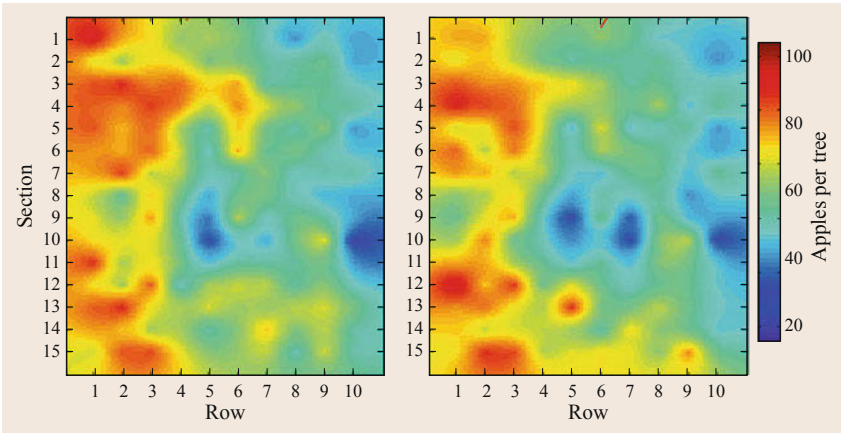


Fig.56.10a,b Spatial yield map collected by the system: **(a)** Truth manual count, **(b)** automated count

Table 56.1 Irrigation improvements at a greenhouse by using a wireless sensor network system (after [56.32])

Condition	Irrigations per day
Before WSN system installed (baseline)	8
With WSN system (monitoring)	4–5
With WSN system (monitoring and control)	2–3

tion also allows growers the ability to monitor and analyze the crop growing environment for trends and long-term changes.

Early results (Tab. 56.1) show water savings of up to 75%, fertilizer leaching reduced to near zero, increased crop quality, speed of crop growth increased by over 50%, and reduced occurrence of crop disease. In addition to these savings, growers can use the data from WSN systems to tailor their crop for specific markets. For example, by adjusting the irrigation set-point value, a grower can choose between more expensive (Grade A) product, or less expensive (Grade B) product that they can sell more of.

56.3.6 Tree Fruit Production

Autonomous orchard vehicles will radically transform tree fruit production by automating maintenance operations such as mowing and spraying and augmenting workers pruning, thinning, and harvesting. From a robotics point of view, these applications can be realized with a relatively simple, yet challenging, capability: driving along one row of trees, turning at the end of the row, and entering the next one. Challenges involved include reliably sensing the trees in the presence of sloped terrain, branches, tall grass, and missing trees; localizing the vehicle in the orchard; following trajectories inside and outside the rows; and avoiding

Table 56.2 Control modes enabled by the family of autonomous orchard vehicles in Fig. 56.11, from least to most complex

	Mule mode	Scaffold mode	Pace mode
How it works	The vehicle drives along rows of trees as workers harvest fruit, placing them in bins on the vehicle	Farm workers stand on the vehicle while it self-steers in the row	The vehicle autonomously drives an entire block at a time without requiring any further interaction
Production tasks enabled	Fruit harvesting	Pruning, fruit and blossom thinning, tree maintenance, harvesting, pheromone dispenser placing	Mowing, spraying, and scouting for disease, insects, and crop yield estimation
Autonomous functionalities	Row following (continuous or *stop-and-go*), end-of-row detection, obstacle detection		Row following, end-of-row detection, turn and enter new row, obstacle detection
Vehicle speed	0.2–2 mph	0.1–0.2 mph	2–5 mph
Interface	Control box with buttons	Control box and foot pedals	Handheld tablet or smartphone
Permanent infrastructure installed in orchard	None	None	None



Fig. 56.11 A family of autonomous orchard vehicles capable of driving among tree rows, based on the Toro Workman series. To keep cost low, the vehicles use only a laser rangefinder and steering and wheel encoder for perception and navigation



Fig. 56.12 **(a)** Autonomous orchard vehicle. **(b)** Workers on foot harvest apples and deposit them on bins towed by the vehicle, eliminating the need for and the cost of a tractor driver. **(c)** In this version the vehicle is equipped with a lift platform from where workers can prune, thin, tie trees, and place pheromone dispensers up to twice as fast as workers performing the same operations on ladders. Users on the platform control the vehicle from an interface **(d)** that communicates with the onboard computer wirelessly

obstacles. Additionally, the added cost of the autonomy components should be as low as possible, to make such vehicles commercially viable.

Figure 56.11 shows a family of autonomous orchard vehicles with a common sensing and computing infrastructure that allows them to cover entire orchard blocks continuously for several hours (VIDEO 91). To keep cost low, they are not equipped with GPS/GNSS or inertial navigation systems; instead, perception and navigation is achieved with a laser rangefinder mounted on the front of the vehicle and steering and wheel encoders. Table 56.2 summarizes the three autonomous control modes in which the vehicles can operate.

From 2008 to 2012 the vehicles drove a combined 350 km in experimental and commercial orchards, vineyards, and nurseries in several US states, including many in the largest apple producer in the United States, Washington state. The longest run covered 25 km over five hours [56.10, 33]. Time trials conducted by The Pennsylvania and Washington State Universities Extension educators showed that workers onboard an autonomous orchard platform can be twice as fast as workers using ladders or on foot when working on the top portion of apple and peach trees (Fig. 56.12) [56.34].

56.3.7 Vehicle Formation Control

Multiple robotic agricultural vehicles driving in formation offer capabilities beyond those possible by a single vehicle. *Lenain* et al. [56.35] developed a control architecture that enables accurate and stable control of several robots in a given and possibly variable formation. The method is based on a path tracking framework, defining the relative robot's positions as a lateral distance with respect to a given path, and the longitudinal distance along this path. Perturbations due to motion on natural environments (e.g., poor grip conditions and uneven terrain topography) are compensated with a nonlinear model-based controller that includes an observer for wheel sideslip. In the formation control point of view the lateral error is no longer regulated to zero but to a desired set-point, potentially variable and accounting for other vehicle deviations. In addition, the velocity of each vehicle is controlled in order to ensure a desired curvilinear distance with respect to the others, and imposing a desired speed for one of them. Similarly to lateral dynamic, the desired distance may be defined as varying, and error may be constituted of a mix between longitudinal errors of one vehicle with respect to the other.

This approach was tested in the field with two electrical off-road robots (Fig. 56.13a). They are equipped with RTK-GPS providing an accuracy of ± 2 cm, and are able to communicate their positions among each other using wireless communications. Weighing around 500 kg each, these two-meter long vehicles are able to reach speeds of $4\,\mathrm{m\,s^{-1}}$ and climb up to 20 degrees terrain. In this example, the first vehicle has to follow the trajectory depicted in black, composed of a straight line along a 15 degree slope on grass at a speed of $2\,\mathrm{m\,s^{-1}}$, followed by a half turn coming back on flat ground. The second vehicle must follow the first one with a desired curvilinear distance of 10 m and a desired lateral

Avidbots Ex1016
Page 1494 of 2154

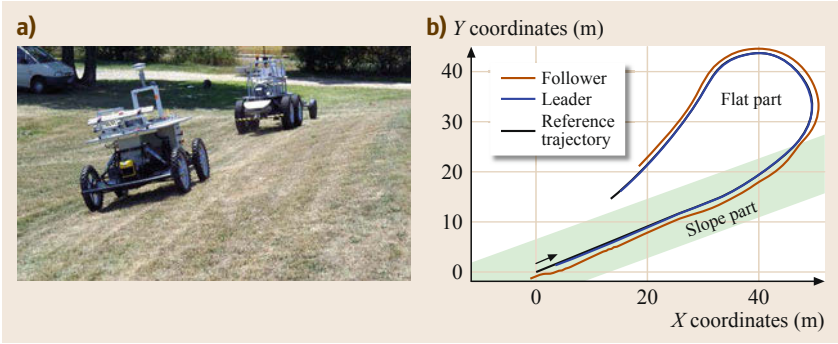


Fig. 56.13 **(a)** Two agricultural vehicles in formation control. **(b)** Reference trajectory and actual path achieved by the two vehicles



Fig. 56.14 Date tree tracking system adapted to a commercial sprayer. The system aims the sprayer nozzle toward the trees more efficiently and safely than a human operator

deviation of 1 m. The positions of the first and second vehicles are shown in blue and red dots in Fig. 56.13b, respectively. In these conditions, the accuracy of relative positioning is within ±15 cm despite the challenging terrain conditions (low grip, uneven terrain, curved trajectory). This work demonstrated the capability of maintaining a formation of several vehicles, compatible with the conditions encountered in agriculture, allowing to consider the introduction of multi-vehicle platoons in the field for precision agriculture.

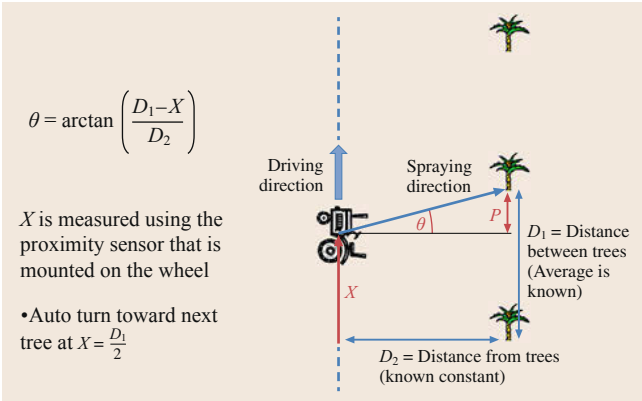


Fig. 56.15 Calculation of the desired spray angle

56.3.8 Date Palm Tree Spraying

Dates palm tree spraying is normally done manually by a team of three workers from a platform 18 m or more above the ground. In the past, many accidents have occurred due to instability of the platform when in a lifted position. Degania Sprayers Company (Israel) developed a sprayer with a tall air cannon and a pan-tilt unit at the end to control the air flow and spray direction. This system, however, requires a worker to manually aim the spray toward the tree. *Shapiro* et al. [56.36] developed an automatic tree tracking system for the sprayer (Fig. 56.14).

The tracking system is based on an ultrasonic range sensor to detect the trees and a proximity sen-

sor mounted on the sprayer's wheel to measure distance traveled. A human driver is responsible for maintaining the sprayer a distance $D_2 = 3.5\,\text{m}$ from the trees. Knowing the average distance between trees in the row ($D_1 = 9\,\text{m}$), it is possible to compute the desired spraying angle as $\theta = \tan^{-1}((D_1 - x)/D_2)$, where x is the distance traveled since the last tree (Fig. 56.15). This value is fed fed into a proportional–integral–derivative (PID) controller that outputs a pulse-width modulation (PWM) signal to an electrically-controlled hydraulic valve that controls the spraying angle. The authors opted to use an on/off valve driven by a PWM signal instead of a proportional valve to reduce the system cost and therefore make it more attractive to farmers. The rotation angle of the sprayer is measured by a potentiometer and used as feedback for the PID controller. When the sprayer reaches the midpoint between two trees, i. e., $x = D_1/2$, the sprayer is set to rotate toward the next tree in the row and start spraying it. The tracking algorithm is implemented on an Arduino microcontroller.

The system was built and deployed in date farms, showing good tracking results. It replaces a human worker which previously had to perform a dull task controlling the sprayer to track the trees using a joystick. A preliminary economic analysis indicates that the additional cost of the sprayer tracking system can be recovered in one harvest season.

56.3.9 Plant Probing

In large botanic experimentation fields, treatments (watering, nutrients, sunlight) that optimize certain desired aspects (growth, appearance) need to be determined. Towards this aim, experiments entailing many repetitive actions need to be conducted. For example, measurements and samples from leaves must be taken regularly and some pruning may need to be performed. For these tasks robots would be very handy, but difficulties arise from the complex structure and deformable nature of plants, which do not only change appearance through growing, but whose leaves also move on a daily and sometimes hourly basis. Even though recent advances in depth sensors, deformable object modeling, and autonomous mobile manipulation have brought this goal in reach for robotic applications [56.37, 38], many problems still exist, in particular concerning robust recognition and localization of plant parts (leaves, flowers, fruits, stems), and robot manipulation under weakly constrained conditions in natural environments.

In this context, the European project GARNICS (gardening with a cognitive system) aims at 3-D sensing of plant growth and building perceptual representations for learning the links to actions of a robot gardener.

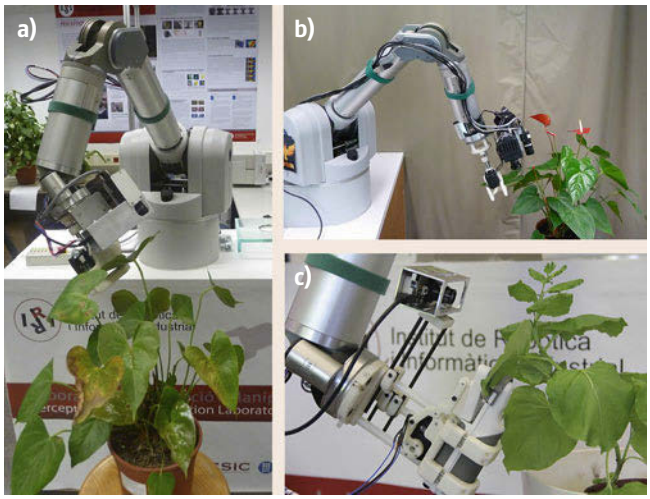


Fig. 56.16 (a) Chlorophyll measurements with a robot-mounted spad meter. (b) Cutting of leaf disks with a robot-mounted, custom-designed cutting tool. Both a color camera and a ToF camera are mounted on the end-effector of the arm. (c) Close-up of probing: Chlorophyll measurements of a tobacco plant. The lightweight time-of-flight camera is seen at the top

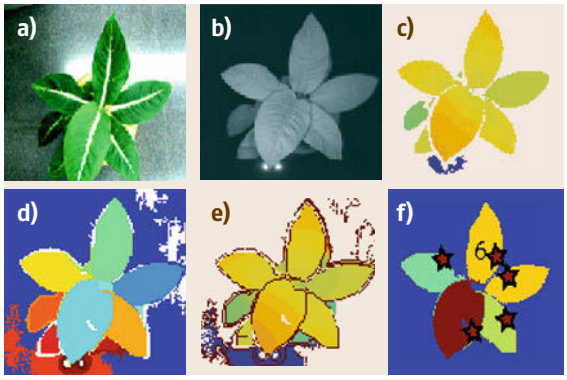


Fig. 56.17 (a) Color image. (b) Infrared image. (c) ToF depth. (d) Segmentation. (e) Surface models. (f) Color-coded leaf confidences and extracted probing points

Plant sensing and control is addressed by combining active vision with appropriate perceptual representations, which are essential for cognitive interactions (VIDEO 95).

An application of robotized phenotyping related to this project is the accurate placing of a measurement tool on a leaf in order to either cut sample disks from the leaf, or to measure chlorophyll content. The robotic arm is equipped with a time-of-flight (TOF) camera and a measurement tool (Fig. 56.16) [56.39–41]. In this approach, image segmentation and model fitting are employed to recognize and localize single leaves from depth information. 3-D data is combined with color or

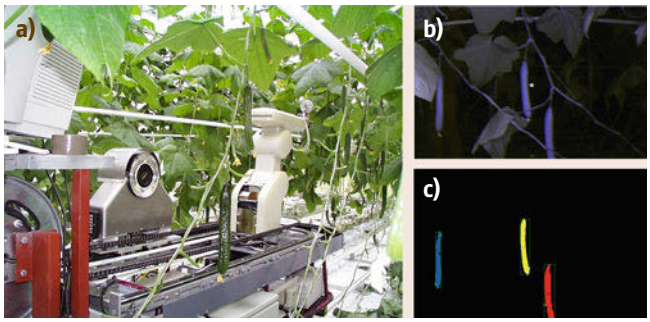


Fig.56.18a–c A cucumber harvesting robot. **(a)** The robot in the greenhouse. **(b)** Unprocessed image of cucumber crop in the near infrared spectrum. **(c)** Detected cucumbers

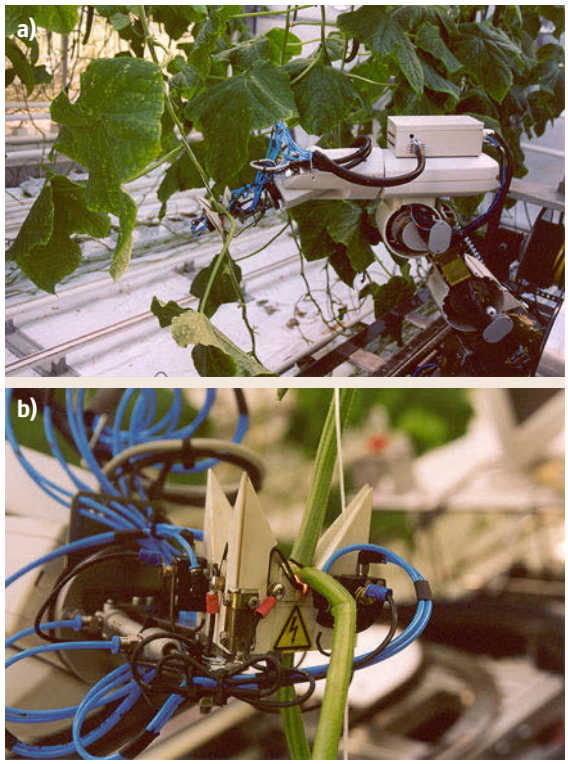


Fig.56.19a,b Cucumber leaf removal robot. **(a)** Robot operational in the crop. **(b)** End-effector (after [56.16])

infrared images and used to segment the data into surface patches, which are assumed to correspond to actual plant leaves [56.42].

In this approach, a next-best view strategy was proposed for finding a non-obstructed and frontal view of the leaf [56.41]. Initially, the robot arm is moved to a position from which a general view of the plant is obtained. The depth and infrared images acquired from this position are segmented into their composite surfaces, leaf model contours are fitted to the extracted segments, and the validity of the fit and the graspability of the leaf are measured (Fig. 56.17). A target leaf is selected and the robot moves the camera to a closer, frontal view. The validity of the target and the graspability are re-evaluated. If the leaf is considered to be suitable for sampling based on these criteria, the probing tool is placed onto the leaf following a two-step path. If the target is not considered suitable for probing, another target leaf (from the general view) is selected and the procedure is repeated.

The method is based on several assumptions:

i) The boundaries of leaves are visible in the infrared-intensity image
ii) The leaf surfaces can be modeled by a basic quadratic function
iii) Leaves of a specific plant type can be described by a common 2-D contour
iv) Leaves are large enough to allow analysis by a ToF camera, and
v) The leaves are static during probing.

These assumptions may be violated under certain conditions, and further research will have to be undertaken to solve the various problems originating mostly from the complex and deformable nature of plants.

56.3.10 Cucumber Harvesting

A prototype robotic harvester for cucumbers is presented in Fig. 56.18. The machine consists of a mobile platform that runs on rails (VIDEO 308). These rails are commonly used in greenhouses in The Netherlands for the purpose of internal transport, but they are also used as a hot water heating system of the greenhouse. Harvesting requires functional steps such as the detection and localization of the fruit and assessment of its ripeness. In case of the cucumber harvester, the different reflection properties in the near infra-red spectrum were exploited to detect green cucumbers in the green environment [56.43–45]. Whether the cucumber was ready for harvest was identified based on an estimation of its weight. Since cucumbers consist almost 95% of water, the weight estimation was achieved by estimating the volume of the fruits. Stereovision principles were then used to locate the fruits to be harvested in the 3-D environment. For that purpose the camera was shifted 50 mm on a linear slide and two images of the same scene were taken and processed. A Mitsubishi RV-E2 manipulator was used to steer the gripper-cutter mechanism to the fruit and transport the harvested fruit back to a storage crate. Collision-free motion planning based on the A* algorithm was used to steer the manipulator during the harvesting operation [56.44]. The cutter consisted of a suction cup on a parallel gripper

that grabbed the peduncle of the fruit. (The peduncle is a stem segment that connects the fruit to the main stem of the plant.) Then the action of a suction cup immobilized the fruit in the gripper. A special thermal cutting device was used to separate the fruit from the plant. The high temperature of the cutting device also prevented the potential transport of viruses from one plant to the other during the harvesting process. For a successful harvest this machine needed 65.2 s on average. The success rate was 74.4% [56.43].

56.3.11 Cucumber Leaf Removal

Harvesting has received considerable attention in robotics research focused on protected cultivation. Yet, this is not the only time- and labor-consuming cultivation operation. In cucumber production, amongst others, removal of old non-productive leaves in the lower regions of the plant is a time-consuming task. Figure 56.19 shows the same platform used for harvesting, but in this case hardware and software were used for removal of leaves from the plants [56.16] (VIDEO 309). In this system, the camera system is used to identify and locate the main stem of the plants. The gripper is sent to the plant and moved upwards. Leaves encountered during this upward motion are separated from the plant using the similar thermal cutting device as used for harvesting. An interesting feature of this machine is that with slight modifications of software and hardware, two greenhouse operations could be performed.

56.3.12 Rose Harvesting

In recent years a harvesting robot for roses has been developed and tested under practical circumstances in The Netherlands. The prototype is shown in Fig. 56.20. In this case, the rose plants are grown on moveable benches. Thus plants move to the robot instead of the robot moving to the plants in the greenhouse. During the harvest cycle, a camera system travels over the rose plants and locates the roses to be harvested. Then the harvesting operation is performed with two manipulators. One manipulator grips the rose just below the flower and pulls it gently aside to generate space for the second manipulator to travel down the stem towards the point where it will cut the stem. This manipulator carries a small-sized stereovision system that is used for real-time tracking of the stem during this downwards motion. Upon arrival, the manipulator deploys a small scissor-type cutter that will cut the stem. Then, finally, the rose is pulled out and put in a storage by the first manipulator whilst the second manipulator moves out of the crop and proceeds with the next harvest cycle. Details of this system have been covered in [56.13].

56.3.13 Strawberry Harvesting

In Japan, the market of strawberries is large, as large as the market of tomatoes, cucumbers, and mandarin oranges. The potentially high-economic return of this product together with the high labor intensity of processes like harvesting, explains the long tradition of research on robotic strawberry harvesting. As shown in Fig. 56.21, the robot consists of a 4 degrees-of-freedom (DOF) cylindrical manipulator. The robot carries 3 charge-coupled device (CCD) cameras. A square LED-array is used for illumination of the scene. Two cameras provide stereo vision for detection and localization of the fruits. Once a fruit is detected, the end-effector is positioned in front of the fruit. The third camera, mounted on the end-effector, is then used to detect the peduncle, i. e., the fruit stem, and calculate its inclination. Based on this data, the orientation of the end-effector is modified with a tilt mechanism and it then approaches the fruit. A successful approach of the strawberry is detected by a reflection-type light sensor in the end-effector. Upon successful completion of this motion, the peduncle is grasped and the stem is cut with a scissor-type cutting mechanism. A suitable manipulator motion then sends the harvested strawberry to a tray. This procedure is repeated for all detected fruits at the



Fig.56.20a,b A rose harvesting robot. **(a)** The robot with the rose crop on a mobile growing system. **(b)** The end-effector with stereo vision system and a scissors-type cutting device (courtesy Jentjens Machinetechniek BV, Veghel, The Netherlands)

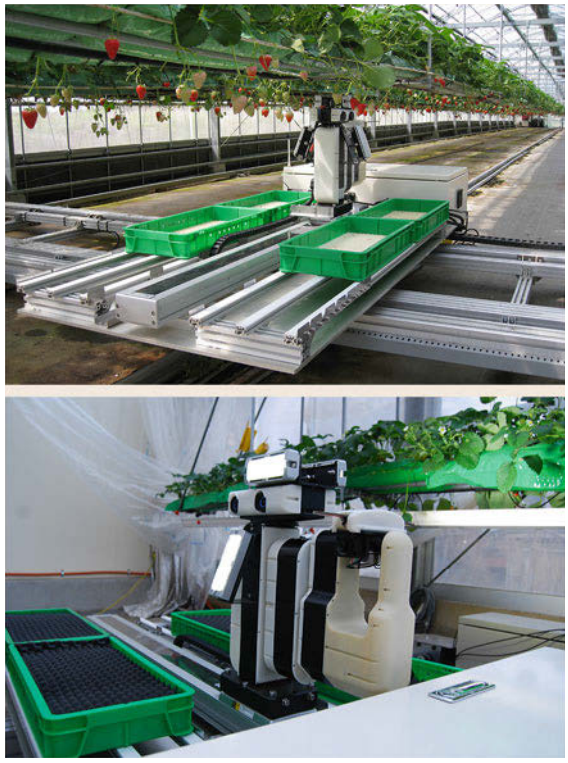


Fig. 56.21 A strawberry harvesting robot (courtesy IAM-Brain, Japan)

current position of the robot. After all picking attempts are completed the full robot platform is moved 210 mm with a gantry-type transportation system running below the strawberry benches. The current prototype has achieved a picking speed of 6.3 s with a success rate of 52.6%. For more details refer to [56.46, 47].

56.3.14 Pot Handling in Nurseries and Greenhouses

Nursery and greenhouse (N&G) farms in the United States produce over two billion potted plants annually (Fig. 56.22a). During the course of production plants are moved several times – distributed onto indoor or outdoor growing beds, repositioned to recover space as orders are filled, and collected for bulk transport. Until recently, only scarce manual labor was suitable for these jobs (Fig. 56.22b).

Harvest Automation's recently-marketed HV-100 (*Harvey*) robot automates critical N&G tasks (Fig. 56.23a). Plants are lifted and transported using a one degree-of-freedom manipulator coupled with a one degree-of-freedom gripper. The mobility system employs two differentially-controlled drive wheels balanced by a front roller.

Harvey uses a laser rangefinder to identify the containers in which plants are grown. This sensor has a horizontal field-of-view greater than 180 degrees and can detect poorly reflecting plant containers from at least 4 m – even under bright sunlight. The robot also uses the laser rangefinder to detect obstacles and robot teammates.

Four *boundary sensors*, two forward-pointing, two rear-pointing, are used to find and follow retroreflective tape that marks the edge of the bed. The tape marker performs double duty as both the robot's global reference and as part of the user interface. By positioning the boundary marker workers indicate to the robot where plants should be placed.

A user interface consisting of a dial and buttons is located on the back of the electronics box. The interface enables users to input the desired plant spacing, bed width, spacing pattern (hexagonal or rectangular) and the number of aisles the robot should instantiate.

Robust operation of the robot is enabled by a behavior-based programming scheme. Figure 56.23b illustrates *spacing*, a task that gives each plant sufficient room to grow without interfering with its neighbors. The closely packed plants in the foreground have been delivered to the growing bed and placed on the ground. Robot A identifies plant containers using the rangefinder; it selects for pick up the container farthest down field. After closing on and capturing a container with its gripper the robot will turn toward the boundary B. Nearing the tape marker the two forward boundary sensors on the robot will detect the tape and compute the relative angle between robot and tape. This enables



Fig. 56.22a,b Potted plants destined for garden stores and landscapers are grown in containers on immense nursery farms. Large amounts of stoop labor is needed to place, maintain, and retrieve these plants

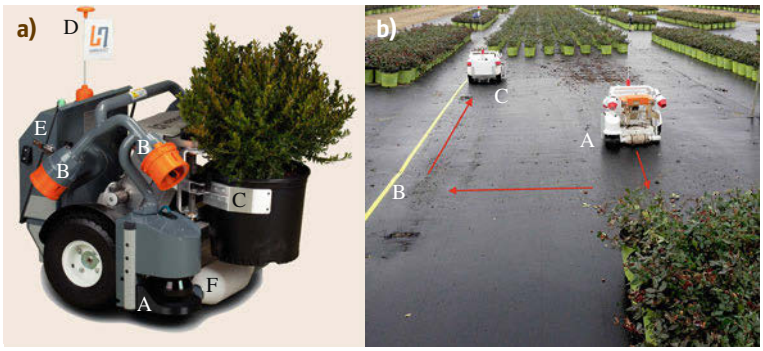


Fig. 56.23 **(a)** HV-100 (*Harvey*) nursery and greenhouse automation robot with main components labeled: (A) laser rangefinder, (B) boundary sensor (two of four), (C) gripper, (D) emergency stop flag (pulling the flag stops the robot), (E) electronics box, with user interface components mounted on the back, and (F) passive roller to balance the robot. **(b)** Two robots operating as a team perform the spacing task. Robot A approaches the source plants in the foreground. Boundary B marks the edge of the bed. Robot C carries a plant toward the destination

the robot to turn and align with the tape, keeping the tape on its left side.

After acquiring the boundary marker the robot will servo along the tape as robot C is doing. When the plant destination comes within view the laser rangefinder is used to identify the next empty space in the pattern of *spaced* plants. The robot then computes an efficient path to the put-down point, moves to that position, and drops off the plant. Afterward, it turns back toward the source of plants and repeats the process. The robot uses dead reckoning to find its way to the vicinity of the source plants. Following this strategy robots can work singly or in teams of various sizes.

56.3.15 Precision Forestry

About 30% of the land mass of the earth is covered by forests. In addition to providing the raw materials for furniture, paper, clothing, and heating, forests provide habitats to diverse animal species and form the source of livelihood for many different human settlements. In 2012, in Germany alone, an estimated 1.3 million jobs in the wood processing industry generated a revenue of more than 180 billion euros. To explore forests in an environmentally-friendly but still economical way is thus a major issue. Robotics is currently being used to preserve the forest and secure the jobs in forestry and related industries.

Today, work in forests is already highly mechanized, and in the last decade, mobile robotics know-how combined with new virtual reality and remote sensing techniques paved the way for a new *robotics view* onto work machines in the forest. Wood harvesters (Fig. 56.24) and forwarders, advanced work machines for log cutting and transport, are currently a major aim of automation efforts [56.45, 48, 49]. Based on the insight that accurate machine localization cannot solely be based on GPS, mobile robotics capabilities for localization and navigation have been introduced. Measured GPS errors of up to 50 m due to signal absorption in the canopy and to multi-path effects make GPS practically useless for precise localization and navigation. In



Fig. 56.24 **(a)** A simulated wood harvester equipped with simulated laser scanners in a virtual testbed used for algorithm development and evaluation. **(b)** The physical harvester in the woods, equipped with laser scanners mounted to the right and left door of the cabin

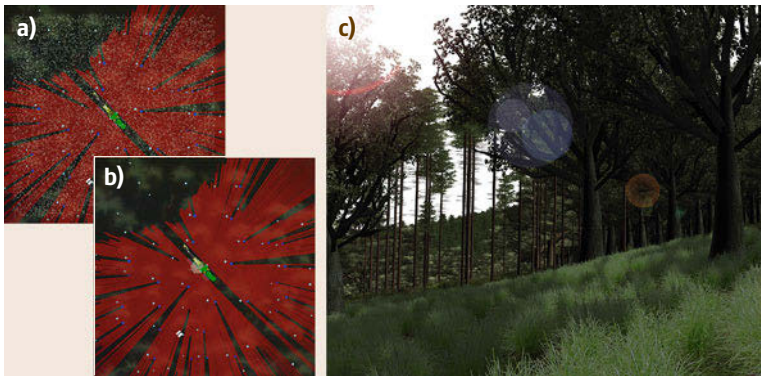


Fig. 56.25 **(a)** Top-down view of the machine, particles (*gray dots*), trees from global tree map (*light blue dots*) and trees from local tree map (*dark blue dots*). **(b)** Particles accumulate near the left edge of the cabin of the forest machine and indicate the newly calculated position in comparison to the GPS-based position (*white rectangle* below the harvester). **(c)** A semantic world model turned into a *virtual forest.*

the *VisualGPS* approach, the GPS position only serves as a starting point for a combined Kalman filter and Monte Carlo localization algorithm based on optical range measurements by laser scanners (VIDEO 96). The approach yields a machine position with an accuracy of 0.5 m and thus provides a sound basis for the development of navigation and (semi-) autonomous logging procedures [56.50, 51].

Practical experiments show that map building based on simultaneous localization and mapping (SLAM) techniques [56.52] are not very applicable in these environments, because the resulting errors in the map are not bounded and thus are not well-suited for large area operation and for a matching against parcel borders. Instead, the highly accurate position estimation from *VisualGPS* builds on a previously generated map of single trees. A multi-sensor fusion approach helps to build this map based on airborne and satellite imagery in multiple spectral ranges as well as on airborne laser scanning. The multispectral imagery data provides the basis for the tree species determination which is solved as an advanced pattern classification problem [56.53]. The next step, single tree delineation, is based on airborne imagery in combination with 3-D surface data from the laser scans. A modified watershed algorithm [56.54] delineates the tree crowns. From the tree crowns' sizes and species information, the geo-referenced stem position and even the trees' diameters can be deduced. This not only allows for generating a global, geo-referenced tree map, but is also sufficient for generating a semantic world model, the *virtual forest* (Fig. 56.25).

Each work machine in the forest also builds local maps of visible trees using machine-mounted laser scanners and a compass. For localization purposes, these local maps are matched against the previously generated global map of trees by means of a particle filter. For a moving machine, the prediction step is carried out by a Kalman filter [56.50, 51].

Figure 56.25 shows the two important results of the global map building process. The figures on the left show just the tree map with the white *noise* in the upper image denoting the particles representing potential poses for the Monte Carlo localization process. In the lower image, the particles have converged against the machine's correct position. The right image gives an impression of the *virtual forest* generated from the same data that was used for the tree map. This *virtual forest* is used to visualize the deduced information (e.g., tree species, tree height, crown shape and size, etc.) in a high-end virtual reality representation. This approach follows a general trend in robotic applications for natural or space environments: the development of virtual testbeds for world model information visualization in an intuitively comprehensible manner. Advanced virtual testbeds then provide the simulation capabilities to develop and test the robotic application against the virtual world in order to save time and costs [56.55].

The developed laser rangefinder-based *VisualGPS* algorithms are being enhanced by optical stereo image recognition capabilities and are being ported to mobile platforms like the Seekur Jr. This is due to the fact that the work that started as an automation effort for work machines in the forest has recently turned into an environment monitoring project which aids the forester to inspect, protect, and attend to the forest with increased efficiency and effectiveness.

56.3.16 Semi-Automation of Forwarder Crane

In cut-to-length harvesting, which is the predominant tree harvesting method in Europe, delimbed and bucked timber is collected from the forest to the roadside by a forwarder. For an operator of such a machine, a large part of the work cycle consists of maneuvering the onboard hydraulic crane (Fig. 56.26). The operator controls the links individually using two joysticks. The redundant kinematic design is necessary for dexterity and the large active workspace of the machine, but it

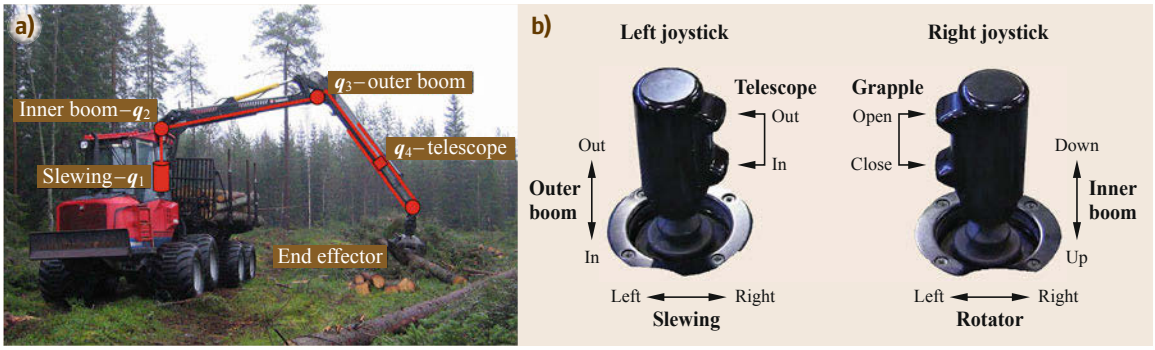


Fig. 56.26 **(a)** Valmet 830 forwarder with redundant kinematic design of the hydraulic crane. **(b)** The operator uses two joysticks to control each link individually

also makes the crane control difficult to learn and to perform in an efficient way.

Considering the technological progress and capabilities of today's forestry machines, the current manual control approach creates bottlenecks in the forwarding process. Semi-automation of these operations would thus be beneficial for productivity. Automation of some repetitive motions is also desired in order to support the operator and to reduce the physical and cognitive workload.

To this end, a small-scale hydraulic forwarder crane has been installed in a laboratory environment [56.56]. The crane is equipped with position and pressure sensors, as well as with electronics and software for rapid prototyping of automated control strategies. The same equipment has been installed on a commercial Valmet 830 forwarder by Komatsu Forest for field testing.

Using these platforms, new feedback control methods and trajectory planning procedures have been developed to construct and implement time-efficient motions [56.57]. For a given geometric path, the speed and relative use of the different crane links along this path can be shaped to achieve motions with optimal performance within the limitations of the machine. Velocity constraints for the individual joints are particularly restrictive in hydraulic manipulators.

In [56.58], trajectories planned with this approach are compared with recorded motions by professional human operators with respect to execution time. Results show that the performance can be improved significantly by path-constrained replanning of human-operated motions (Fig. 56.27). Additional replanning of the geometrical path, along with efficient velocity profiles along the path, can further improve the time efficiency of the crane motions.

Tracking the trajectories with feedback control requires sensors for measuring the joint positions. Installation of such sensors may not be desired due to high costs or durability problems in the rough outdoor environment. Using the approach in [56.59], motions can be found that are robust to uncertainties in initial conditions and therefore possible to perform in open loop. The end effector position along the trajectory can also be estimated using an observer and signals available from pressure sensors. Such design can be a more convenient solution in the harsh conditions of the forest.

The introduction of new tasks or new ways to perform tasks means new challenges for human-machine interaction (HMI). Semi-autonomous operation requires the automated parts to be well integrated with the manual work. In order to facilitate cooperation in manipulator control between the human and computer, the work distribution and work transitions between them is important. *Hansson* and *Servin* [56.60] describes an implementation of shared and traded control for this setup. The results from user tests show that semi-autonomous operation reduces the workload and shows significant potential to improve the productivity of inexperienced operators.

A reliable framework of sensors and low-level control, along with efficient motion planning strategies, allows for development of more advanced interaction technologies. One such possible future scenario is teleoperation, that presents several advantages to both machine owners and operators. Firstly, it opens up for redesign of the machine with removal of the cabin which saves weight and cost. Secondly, the working environment is improved with reduction of the noise and vibration levels. A virtual environment-based system for teleoperation of a forwarder crane was demonstrated in [56.61]; follow-up work is investigating how a virtual environment can be a useful tool for presenting feedback to the operator in different scenarios. This method does however require environment reconstruction to locate logs and potential obstacles, both of which are considered in [56.62].

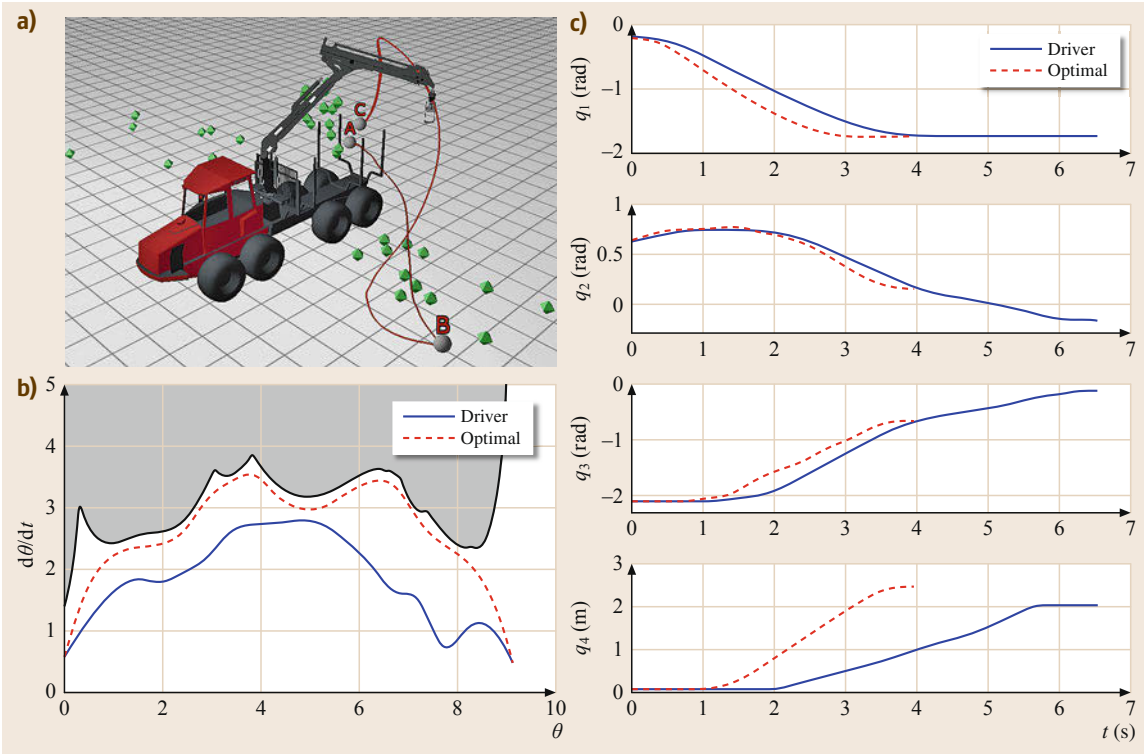


Fig.56.27a–c Path-constrained replanning of human-operated motions. **(a)** Example of two crane motions performed by a human operator: from the load bunk to the logs (A–B) and back (B–C). **(b)** Velocity profiles for the distance θ along the path A–B. A different distribution of the work between the joints allows for a higher velocity along the same path. The *gray area* indicates velocity constraints not to be violated. **(c)** The resulting evolution of joint coordinates with respect to time. The motion with a time-efficient velocity profile is clearly faster than the one of the driver

56.3.17 Livestock Breeding and Nurturing

In some cases animal breeding is a proactive process, such as in hatcheries where optimum conditions must be managed. In others nature is allowed to take its course. Sometimes intervention may be needed in the birthing process, but in the Australian outback cattle must take their chances. Nevertheless the newborn must be marked with transponder tags under the national livestock identification scheme, NLIS [56.63].

Other species such as kangaroos, feral pigs, wild camels and horses can breed and run wild until harvested or culled. When creatures are confined, tasks



Fig. 56.28 Lely Juno robot



Fig. 56.29 Lely Vector feeder

include environmental monitoring and control, feeding, cleaning and growth monitoring.

A Dutch company, Lely [56.64], is active in marketing products that have a strong robotic element. When cattle are located in stalls, the Juno [56.65] patrols the laneway at the side of the pen, pushing fodder towards the rails to put it in reach of the feeding cattle (Fig. 56.28).

The Vector [56.66] goes even further. A mobile robot carrying a hopper, it navigates automatically between the barn that holds the feed and the stalls where the cattle are waiting (Fig. 56.29).

Other equipment in Australian feedlots estimates the weight gain of the cattle, such as a system that reads the NLIS tag of a beast at a water trough and records the weight imposed by the front legs.

In the Australian outback, similar sensing systems perform a variety of functions. Water is scarce, so watering points can be fenced, giving access only via a laneway that can be monitored. Some earlier projects [56.67] have involved species identification to control access by means of an automatically operated gate, but similar technology has valuable use in cattle production.

Again the NLIS tag is read, while a *walk over* weighing system notes the weight gain of the animal. Vision can be used to identify cows that are followed by a calf. Now automatic gates can direct these to a separate paddock for tagging the calves.

Many of the opportunities are for automation, which may be a precursor for robotics. Several projects have involved machine-vision monitoring of cattle [56.68] or of pigs in piggeries [56.69]. The weight-gain is estimated by visual means using precision cameras (Fig. 56.30). Other analyses concern behavior [56.70].

Mobile robots can be involved in sheds where chickens are bred for meat, rather than laying. A mobile robot has been proposed for moving among the poultry to monitor air quality including temperature, relative humidity and the concentrations of ammonia and dust.



Fig. 56.30 Measuring the live weight of cattle with vision

Australian honeybees must also be protected from pests, carried by bees on arriving ships. Bait boxes are deployed at ports around Australia to attract any such swarms. A remote monitoring system with camera sensing gives an early alert to the apiary officer [56.71].

56.3.18 Livestock Exploitation

Traditional farming practices milk cows twice a day; in morning and in the evening. Automatic milking parlors with robotic milking stations are being widely adopted to improve diary productivity and convenience (Fig. 56.31). These allow the cows to self-determine when to come in for milking and feeding. In an automated milking station, teats are located and attachment



Fig. 56.31 Robot milking

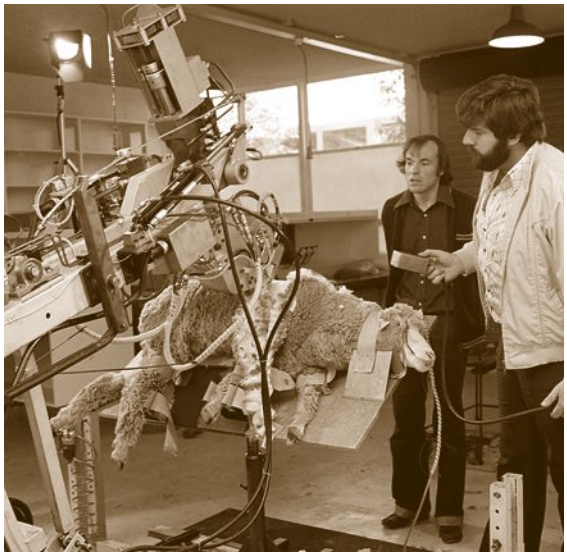


Fig. 56.32 Engineers demonstrate how their pioneering sheep shearing robot works at the University of Western Australia in 1980 (courtesy National Archives of Australia: A6180, 23/9/80/1)

of milking systems is automatic. Yield is monitored and udders are automatically inspected for injury and disease. Human intervention is minimal. The Lely company is active in this area [56.72].

Once a year sheep must be shorn of their wool. In the past, the University of Western Australia developed an automatic shearing system known as the Shear Magic [56.73] (Fig. 56.32). The commercial exploitation was not a success and there seems to have been little activity in this area in recent years.

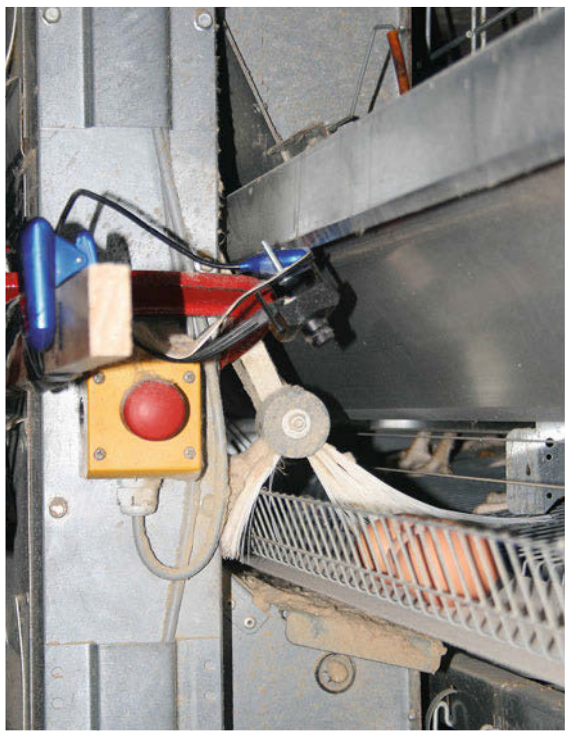


Fig. 56.33 Prototype computer vision system for scanning the egg collection belt for potential blockages



Fig. 56.34 Precision Pastoral's mustering method

Eggs must be collected from battery hens. As they are laid, they roll from the cages onto a conveyer belt. Among other robotic applications in egg production, machine vision is used to detect foreign bodies or damaged eggs [56.74] (Fig. 56.33).

56.3.19 Livestock Harvesting, Slaughtering, and Processing

The system already described for monitoring cattle that approach a waterhole is also the subject of a funded project for mustering. Cattle, selected by means of their NLIS tags, are diverted into a compound from which they can be collected for transport to a feedlot, in preparation for slaughter [56.75] (Fig. 56.34).

The vision-based system that can control access to waterholes can also be used for the collection of feral animals such as pigs. There are said to be more wild camels in Australia than in Arabia and these too can be captured by such a system.

At the start of the 1990s, Fututech was hailed as the future of slaughterhouses [56.76]. Robotic systems to be installed at the Kilcoy pastoral company in Queensland would automate the whole process from the *knocking box* to the chillroom. Construction started in 1992, but by June 1994 the project was abandoned, at a cost of over $40 million [56.77]. It was a project that was ahead of its time, relying on a central computer and kilometers of cabling rather than distributed intelligence.

But much is different now. Robots have become commonplace in abattoirs, supported by sophisticated sensing systems to locate skeletal features [56.78] (Fig. 56.35). Pigs probably outnumber cattle and the processed animals include sheep and goats. Deboning systems are marketed for products including chicken legs and ham bones [56.79, 80].

For slaughtering quantity, the lead must be held by poultry, where a single plant can process up to 4000 birds per hour [56.81].



Fig. 56.35 Lamb processing

Avidbots Ex1016
Page 1505 of 2154

56.3.20 Aerial-Based Precision Agriculture

Unmanned aerial vehicles (UAVs) have recently begun being applied to precision agriculture. This is a new development and results are still preliminary, albeit very promising (VIDEO 307). In fact, the Association for Unmanned Vehicle Systems International forecasts that 80% of all UAVs sold in the United States in the 2015–2025 period will go to serve the agricultural market [56.82]. Even if the actual figure turns out not to be that high, there is no question that roboticists and agricultural engineers are investing significant time and resources to understanding how UAVs can improve agricultural efficiency and reduce costs.

Dong et al. [56.83] use high-resolution images and on-board sensor data from an aerial vehicle (Fig. 56.36) to create a sequence of dense 3-D reconstructions of the crop over time. This four-dimensional (4-D) spatio-temporal reconstruction is used to segment the canopies and estimate how the crown radius and height of each plant evolve. The processing pipeline was tested on data collected in a field planted with corn, broccoli, and cabbage in Tifton, GA, USA (Fig. 56.37).



Fig. 56.36 AscTec Pelican quadrotor used for data collection over early season sweet corn

In many potato growing areas, the potato crop does not senesce naturally and therefore tuber maturation is artificially induced by killing the haulm 10 to 25 days prior to harvest. Reglone is a widely-used potato haulm killing herbicide. Building on the knowledge of the relationship between potato crop biomass status, expressed in weighted difference vegetation index (WDVI) [56.84], and the minimum effective

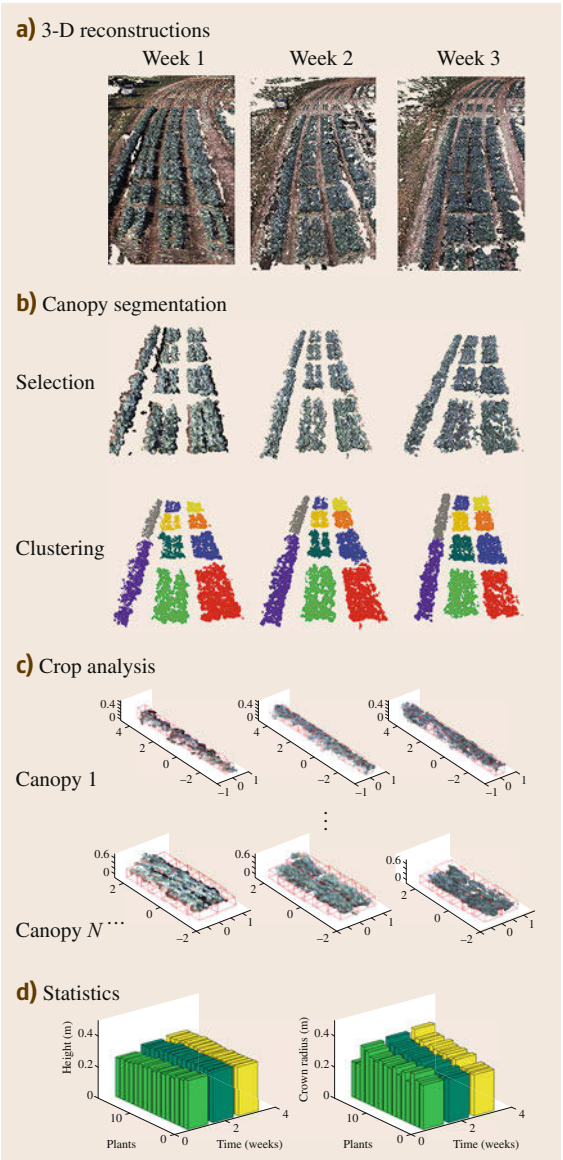


Fig.56.37a–d Dense 3-D reconstruction of the corn stalks, used to estimate how the crown radius and the height of each plant evolve over time. **(a)** 3-D reconstruction of crop, **(b)** canopy segmentation, **(c)** crop analysis and **(d)** crop statistics



Fig. 56.38 Operator launching an unmanned aircraft over a potato field via a launch rope. The plane then flies autonomously following a pre-programmed flight path

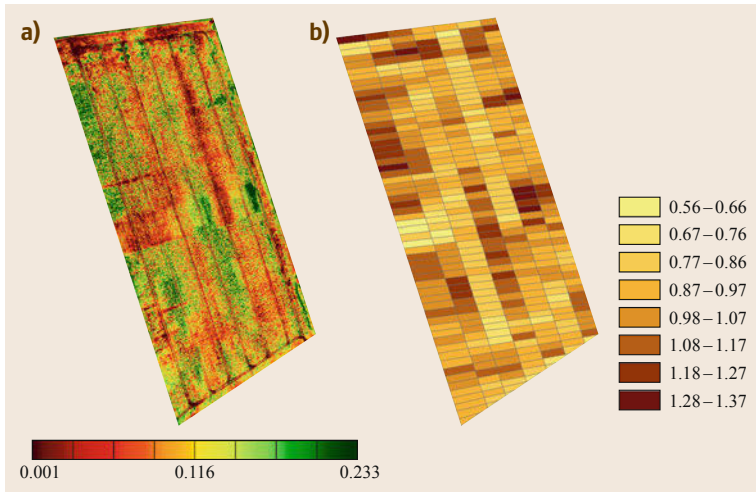


Fig. 56.39 **(a)** Weighted difference vegetation index (WDVI) map of a potato field; the legend on the bottom represents the WDVI, which varies from zero (no crop at all) to one (theoretical maximum). A vigorous potato crop has a value in the 0.6–0.7 range. **(b)** Reglone dose task map; the legend represents dose in liters per hectare

dose of Reglone, *Kempenaar* et al. [56.85] successfully demonstrated the use of crop biomass imaging with a multi-spectral camera underneath an unmanned aircraft (Fig. 56.38) for variable rate application. The WDVI map was converted into a 33 m × 10 m dose grid map (Fig. 56.39) adjusted to the boom width of the field sprayer. On the field, the average use was 0.9 liters of Reglone per ha, with satisfactory efficacy. Standard practice would have been the use of 2 liters per ha. The use of unmanned aerial imaging thus prompted a savings of more than 50%, without loss in potato haulm killing efficacy, harvestability of the potatoes, and final product quality.



Fig. 56.40 Fixed-wing UAV launched over a black-grass-infested wheat field

Wheat is the most widely grown arable crop in the UK, covering around 1.6 million hectares and producing 11.9 million metric tons in 2013. As with other crops, wheat has its share of weed competitors, one of which is black-grass. This prolific weed is highly competitive and increasingly resistant to chemical control, making it one of the biggest challenges facing UK's agricultural sector.

Researchers from URSULA Agriculture have demonstrated a system that combines multispectral sensing with UAVs to capture imagery at 10 cm resolu-

tion (Fig. 56.40) [56.86]. Using the unique spectral and morphological properties of the black-grass, they developed classification processes that are applied to the image to automatically delineate the black-grass from the host crop.

For the farmer and agronomist, the output is supplied in the form of visual maps and spatially attributed data points that are transferable through farm planning software to precision guided machinery (Fig. 56.41). URSULA Agriculture's imagery analysis is able to not only identify the location and in-field extent of black-grass infestation, but also the density and area of black-grass plants, all of which will have a bearing on control decisions. The information provides the farmer with vital data to help contain black-grass infestation in the current growing season. Crucially, it also helps with long-term control by informing management methods such as higher seed rates to increase competition in black-grass *hotspots*, alternative cultivation techniques, or adjusting cropping strategies, all of which can help control black-grass alongside chemical herbicides.

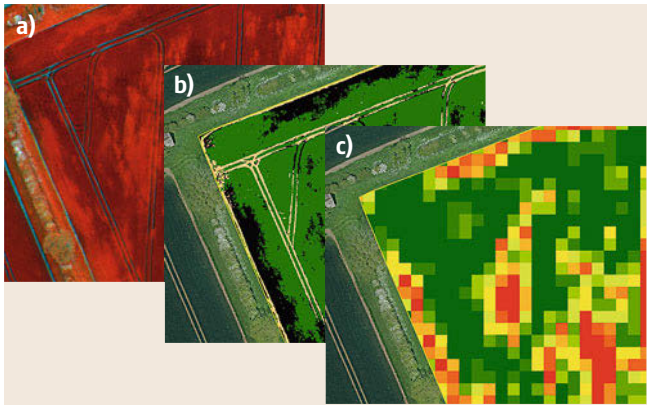


Fig.56.41a–b Black-grass analysis stack: High-resolution multispectral image **(a)**, black-grass delineation **(b)**, and density map **(c)**

56.4 Conclusion

The robotics community has made great strides toward introducing robotic systems in agriculture and forestry. The next ten years will bring improved sensing and mobility, and some advances in manipulation for crop production.

In orchard crops, there is still a need for improvement in algorithms that calculate crop yield and canopy volume or that automatically detect insects and disease. Here the breakthroughs may end up coming from physicists and engineers producing novel imaging systems rather than from roboticists. The continued conversion of existing orchards to the *tree wall* architecture means that autonomous orchard vehicles and platforms will soon have access to the vast majority of the fruit production acreage, at least in developed countries.

Manipulation in orchard environments is a growing and challenging field. The long-sought *harvesting robot* is still an object of fiction, especially when it comes to its economic feasibility. In the apple industry, for example, the best human workers are capable of picking 40 to 60 pieces of fruit per minute, keeping bruising down to a few percent of the volume picked. Our community is relatively far from achieving this level of performance, but we do not need to get there in one shot. Many value-added operations, including pruning, thinning, and harvesting, could benefit from an automated manipulation-based solution. Manipulation should be introduced into low-precision tasks first, where economic gains can be more readily realized, before the technology is refined and directed at tasks such as apple picking. One example is automatic grape shoot and fruit thinning according to each individual vine's crop load [56.87]. This is in contrast to today's methods, where crop load management is conducted uniformly over the entire vineyard, leading to significant variations in yield. Achieving vine-level manipulation in an economically sound way would help move the grape industry in the direction of variable rate management, so that one day, each vine, leaf, and fruit will be treated individually according to its own needs of water, nutrients, light, etc. [56.88].

Despite more than thirty years of research, robotic systems for plant maintenance operations in protected cultivation have not become commercially available. Clearly, the complexity of the operation is an issue. This is due to the highly unstructured working environment, the inherent natural variability of the crop, and the unfavorable environmental conditions, e.g., strong variations in lighting. In this domain, robust sensing and perception is key to successful deployments. This includes aspects such as detection of fruits, stems, and leaves, determination of their properties, e.g., ripeness, and, finally, their accurate localization in the 3-D working environment. Object detection can be quite a challenge as in some cases it boils down to finding a green object in a green background. When color differences are more pronounced detection can be easier, but still in many cases object occlusion poses another problem.

This does not only prevent proper detection of the fruits, but also adds difficulty in the proper assessment of their properties.

In the cluttered environments of protected cultivation, manipulation is also quite challenging. While performing a task, the robot should prevent damage to the crop as this will immediately reduce its value. Still, the robot should be allowed to hit objects such as leaves. Humans tend to have quite intensive contact with the crop during operations. To mimic this kind of behavior requires knowledge about whether objects should not be touched at all or whether they might be gently touched and pushed aside, employing a kind of compliant motion.

Generally speaking, robotics-based crop production can be seen as an instantiation of the hand-eye coordination problem, or the integration of effective sensing, perception, and intelligent manipulation; in other words, those aspects of a well-trained human that are hard to reproduce. Development of technology to mimic human behavior is one way to go, but it will always be limited by the environmental conditions found at the time the robot was introduced. A systemic approach where the crop production system is designed top-down with the robotic system as just one of its components is a much more promising route, yet one that is still in its infancy. For example, growing systems can be modified to make detection and the approach of fruits and flowers easier, thus allowing for more simple design and operation of automated systems. Likewise, cultural practices may be modified to include the robot as part of the task [56.89, 90]. Human-robot interaction might be an interesting and cost-effective intermediate step – allowing for human guidance and supervision only when the machine needs assistance. Such collaborative approaches facilitate data collection in real working environments, offering the opportunity for learning and improving algorithms, thus paving the way for fully autonomous operations in the future.

Another important aspect of the successful introduction of robots in agriculture and forestry is that related to the socio-economic barriers to adoption. In addition to being able to execute a task correctly, the robot must do it in a cost-effective way. Such efficiency can only be demonstrated via numerous field trials, which are time-consuming and costly. Furthermore, the robotic system must have a total ownership cost, including acquisition, maintenance, user training, and disposal, that is less than the financial gains leveraged through its introduction in the production process. Finally, one fundamental aspect is safety. Not only must the hardware and software be designed and validated based on explicit safety requirements, there must be standards and regulations that dictate how and when robots and humans can interact. For an example of a study on the socio-economic barriers to adoption in the apple industry, see [56.91].

The mechanization enabled by the Industrial Revolution and the automation enabled by the information technology era have revolutionized agricultural production to the point where a single farmer can produce grains to feed one hundred people. In the next fifty years, we will witness a similar occurrence in fruits, vegetables, and other crops, thanks to the advances in robotic technologies that our community is continually developing for and applying to agriculture and forestry (VIDEO 93).

Video-References

VIDEO 26 Autonomous orchard tractors
available from http://handbookofrobotics.org/view-chapter/56/videodetails/26

VIDEO 91 Autonomous orchard vehicle for specialty crop production
available from http://handbookofrobotics.org/view-chapter/56/videodetails/91

VIDEO 93 Autonomous utility vehicle – R Gator
available from http://handbookofrobotics.org/view-chapter/56/videodetails/93

VIDEO 95 Automatic plant probing
available from http://handbookofrobotics.org/view-chapter/56/videodetails/95

VIDEO 96 Visual GPS – High accuracy localization for forestry machinery
available from http://handbookofrobotics.org/view-chapter/56/videodetails/96

VIDEO 131 Smart Seeder: An autonomous high accuracy seed planter for broad acre crops
available from http://handbookofrobotics.org/view-chapter/56/videodetails/131

VIDEO 304 A robot for harvesting sweet-pepper in greenhouses
available from http://handbookofrobotics.org/view-chapter/56/videodetails/304

VIDEO 305 Ladybird: An intelligent farm robot for the vegetable industry
available from http://handbookofrobotics.org/view-chapter/56/videodetails/305

VIDEO 306 An automated mobile platform for orchard scanning and for soil, yield, and flower mapping
available from http://handbookofrobotics.org/view-chapter/56/videodetails/306

Avidbots Ex1016
Page 1509 of 2154

VIDEO 307 A mini unmanned aerial system for remote sensing in agriculture
available from http://handbookofrobotics.org/view-chapter/56/videodetails/307
VIDEO 308 An autonomous cucumber harvester
available from http://handbookofrobotics.org/view-chapter/56/videodetails/308
VIDEO 309 An autonomous robot for de-leafing cucumber plants
available from http://handbookofrobotics.org/view-chapter/56/videodetails/309
VIDEO 310 The Intelligent Autonomous Weeder
available from http://handbookofrobotics.org/view-chapter/56/videodetails/310

References

56.1 J.A. Foley, N. Ramankutty, K.A. Brauman, E.S. Cassidy, J.S. Gerber, M. Johnston, N.D. Mueller, C. O'Connell, D.K. Ray, P.C. West, C. Balzer, E.M. Bennett, S.R. Carpenter, J. Hill, C. Monfreda, S. Polasky, J. Rockström, J. Sheehan, S. Siebert, D. Tilman, D.P.M. Zaks: Solutions for a cultivated planet, Nature **478**, 337–342 (2011)

56.2 A. Alleyne: Editor's note: Agriculture and information technology, Natl. Acad. Eng. Bridge, Issue Agric. Inf, Technol. **41**(3), 3–4 (2011)

56.3 Global Harvest Initiative: The 2012 Global Agricultural Productivity Report, http://goo.gl/GBPvjK

56.4 J. Reid: The impact of mechanization on agriculture, Nat. Acad. Eng. Bridge, Issue Agric. Inf. Technol. **41**(3), 22–29 (2011)

56.5 E.J. Van Henten: Greenhouse mechanization: State of the art and future perspective, Acta Hortic. **710**, 55–69 (2006)

56.6 S.T. Nuske, S. Achar, T. Bates, S.G. Narasimhan, S. Singh: Yield estimation in vineyards by visual grape detection, IEEE/RSJ Int. Conf. Intell. Robots Syst., San Francisco (2011)

56.7 Q. Wang, S.T. Nuske, M. Bergerman, S. Singh: Automated crop yield estimation for apple orchards, Int. Symp. Exp. Robotics, Quebec City (2012)

56.8 J. Deere: Integrated AutoTrac System, http://goo.gl/CsxxfL

56.9 K. Gallardo, M. Taylor, H. Hinman: 2009 Cost estimates of establishing and producing gala apples in Washington, Washington State University Extension Fact Sheet FS005E, http://goo.gl/BHBZrb (2010)

56.10 B. Hamner, M. Bergerman, S. Singh: Results with autonomous vehicles operating in specialty crops, IEEE Int. Conf. Robotics Autom., St. Paul, MN (2012)

56.11 G. Giacomelli, N. Castilla, E.J. Van Henten, D.R. Mears, S. Sase: Innovation in greenhouse engineering, Acta Hortic. **801**, 75–88 (2008)

56.12 C.W. Bac, E.J. van Henten, J. Hemming, Y. Edan: Harvesting robots for high-value crops: State-of-the-art review and challenges ahead, J. Field Robotics **31**(6), 888–911 (2012)

56.13 J.C. Noordam, J. Hemming, C. van Heerde, F. Golbach, R. van Soest, E. Wekking: Automated rose cutting in greenhouses with 3D vision and robotics: Analysis of 3D vision techniques for stem detection, Acta Hortic. **691**, 885–892 (2005)

56.14 T. Rath, M. Kawollek: Robotic harvesting of Gerbera Jamesonii based on detection and three-dimensional modeling of cut flower pedicels, Comput. Electron. Agric. **66**, 85–92 (2009)

56.15 R. Berenstein, O.B. Shahar, A. Shapiro, Y. Edan: Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer, Intell. Serv. Robotics **3**, 233–243 (2010)

56.16 E.J. Van Henten, B.A.J. Van Tuijl, G.J. Hoogakker, M.J. Van Der Weerd, J. Hemming, J.G. Kornet, J. Bontsema: An autonomous robot for de-leafing cucumber plants grown in a high-wire cultivation system, Biosyst. Eng. **94**, 317–323 (2006)

56.17 M. Monta, N. Kondo, Y. Shibano: Agricultural robot in grape production system, Int. Conf. Robotics Autom. (1995)

56.18 Wikipedia: Controlled Traffic Farming, http://goo.gl/F0DY04

56.19 J. Jin, L. Tang: Coverage path planning on three-dimensional terrain for arable farming, J. Field Robotics **28**(3), 424–440 (2011)

56.20 D.C. Slaughter, D.K. Giles, D. Downey: Autonomous robotic weed control systems: A review, Comput. Electron. Agric. **61**, 63–78 (2008)

56.21 F.K. Van Evert, J. Samsom, G. Polder, M. Vijn, H.-J. van Dooren, A. Lamaker, G.W.A.M. van der Heijden, C. Kempenaar, T. van der Zalm, B. Lotz: A robot to detect and control broad-leaved dock (Rumex obtusifolius L.) in grassland, J. Field Robotics **28**, 264–277 (2011)

56.22 G. Polder, F.K. Van Evert, A. Lamaker, A. De Jong, G.W.A.M. Van der Heijden, L.A.P. Lotz, T. Van der Zalm, C. Kempenaar: Weed detection using textural image analysis, 6th Bienn. Conf. Eur. Fed. IT Agric. (EFITA), Glasgow (2007), available online at http://edepot.wur.nl/28203

56.23 F.K. Van Evert, G. Polder, G.W.A.M. Van der Heijden, C. Kempenaar, L.A.P. Lotz: Real-time, vision-based detection of Rumex obtusifolius L. in grassland, Weed Res. **49**, 164–174 (2009)

56.24 H. Böhm, J. Finze: Überprüfung der Effektivität der maschinellen Ampferregulierung im Grünland mittels WUZI unter differenzierten Standortbedingungen [Testing the effectiveness of mechanical control of docks in grassland with the WUZI under a variety of conditions.], http://goo.gl/BVdqgy (2004)

56.25 A.T. Nieuwenhuizen, J.W. Hofstee, E.J. van Henten: Adaptive detection of volunteer potato plants in sugar beet fields, Precis. Agric. **11**, 433–447 (2009)

Avidbots Ex1016
Page 1510 of 2154

56.26 A.T. Nieuwenhuizen, J.W. Hofstee, J.C. van de Zande, J. Meuleman, E.J. van Henten: Classification of sugar beet and volunteer potato reflection spectra with a neural network and statistical discriminant analysis to select discriminative wavelengths, Comput. Electron. Agric. **73**, 146–153 (2010)

56.27 A.T. Nieuwenhuizen, J.W. Hofstee, E.J. van Henten: Performance evaluation of an automated detection and control system for volunteer potatoes in sugar beet fields, Biosyst. Eng. **107**, 46–53 (2010)

56.28 T. Bakker, H. Wouters, K. Van Asselt, J. Bontsema, L. Tang, J. Müller, G. Van Straten: A vision based row detection system for sugar beet, Comput. Electron. Agric. **60**(1), 87–95 (2008)

56.29 T. Bakker, K. Van Asselt, J. Bontsema, J. Müller, G. Van Straten: A path following algorithm for mobile robots, Auton. Robots **29**(1), 85–97 (2010)

56.30 T. Bakker, K. van Asselt, J. Bontsema, J. Müller, G. van Straten: Autonomous navigation using a robot platform in a sugar beet field, Biosyst. Eng. **109**(4), 357–368 (2011)

56.31 J. Katupitiya, R. Eaton, T. Yaqub: Systems engineering approach to agricultural automation: New developments, 1st Annual IEEE Syst. Conf. (2007) pp. 298–304

56.32 D. Kohanbash, A. Valada, G.A. Kantor: Irrigation control methods for wireless sensor network, Am. Soc. Agric. Biol. Eng. Annual Meeting (2012), available online at http://goo.gl/3NJyXb

56.33 S. Singh, M. Bergerman, J. Cannons, B. Grocholsky, B. Hamner, G. Holguin, L. Hull, V. Jones, G. Kantor, H. Koselka, G. Li, J. Owen, J. Park, W. Shi, J. Teza: Comprehensive automation for specialty crops: Year 1 results and lessons learned, J. Intell. Serv. Robotics, Special Issue Agric, Robotics **3**(4), 245–262 (2010)

56.34 B. Davis: CMU-led automation program puts robots in the field, AUVSI's Unmanned Syst.: Mission Crit. **2**, 38–40 (2012)

56.35 R. Lenain, J. Preynat, B. Thuilot, P. Avanzini, P. Martinet: Adaptive formation control of a fleet of mobile robots: Application to autonomous field operations, Int. Conf. Robotics Autom. (2010) pp. 1241–1246

56.36 A. Shapiro, E. Korkidi, A. Demri, R. Riemer, Y. Edan, O. Ben-Shahar: Toward elevated agrobotics: An autonomous field robot for spraying and pollinating date palm trees, J. Field Robotics **26**(6/7), 572–590 (2009)

56.37 S. Foix, G. Alenyà, C. Torras: Lock-in time-of-flight (ToF) cameras: A survey, IEEE Sensors J. **11**(9), 1917–1926 (2011)

56.38 R. Klose, J. Penlington, A. Ruckelshausen: Usability study of 3D time-of-flight cameras for automatic plant phenotyping, Workshop Comput. Image Anal. Agric. (2009) pp. 93–105

56.39 G. Alenyà, B. Dellen, S. Foix, C. Torras: Leaf segmentation from ToF data for robotized plant probing, IEEE Robotics Autom. Mag. **20**(3), 50–59 (2013)

56.40 G. Alenyà, B. Dellen, S. Foix, C. Torras: Robotic leaf probing via segmentation of range data into surface patches, IROS Workshop Agric. Robotics, Vilamura (2012)

56.41 G. Alenyà, B. Dellen, C. Torras: 3D modelling of leaves from color and ToF data for robotized plant measuring, IEEE Int. Conf. Robotics Autom., Shanghai (2011) pp. 3408–3414

56.42 B. Dellen, G. Alenyà, S. Foix, C. Torras: Segmenting color images into surface patches by exploiting sparse depth data, IEEE Workshop Appl. Comput. Vis., Kona (2011) pp. 591–598

56.43 E.J. Van Henten, B.A.J. Van Tuijl, J. Hemming, J.G. Kornet, J. Bontsema, E.A. Van Os: Field test of an autonomous cucumber picking robot, Biosyst. Eng. **86**, 305–313 (2003)

56.44 E.J. Van Henten, J. Hemming, B.A.J. Van Tuijl, J.G. Kornet, J. Bontsema: Collision-free motion planning for a cucumber picking robot, Biosyst. Eng. **86**, 135–144 (2003)

56.45 E.J. Van Henten, J. Hemming, B.A.J. Van Tuijl, J.G. Kornet, J. Meuleman, J. Bontsema, E.A. Van Os: An autonomous robot for harvesting cucumbers in greenhouses, Auton. Robots **13**, 241–258 (2002)

56.46 S. Hayashi, K. Shigematsu, S. Yamamoto, K. Kobayashi, Y. Kohno, J. Kamata, M. Kurita: Evaluation of a strawberry-harvesting robot in a field test, Biosyst. Eng. **105**, 160–171 (2010)

56.47 S. Hayashi, S. Yamamoto, S. Saito, Y. Ochiai, Y. Kohno, K. Yamamoto, J. Kamata, M. Kurita: Development of a movable strawberry-harvesting robot using a travelling platform, Proc. Int. Conf. Agric. Eng. CIGR-AgEng 2012, Valencia, Spain (2012)

56.48 F. Georgsson, T. Hellström, T. Johansson, K. Prorok, O. Ringdahl, U. Sandström: Development of an autonomous path tracking forest machine – a status report. In: *Field and Service Robotics: Results of the 5th International Conference*, ed. by P. Corke, S. Sukkarieh (Springer, Berlin, Heidelberg 2006) pp. 603–614

56.49 M. Miettinen, M. Öhman, A. Visala, P. Forsman: Simultaneous localization and mapping for forest harvesters, IEEE Int. Conf. Robotics Autom. (2007) pp. 517–522

56.50 J. Roßmann, M. Schluse, C. Schlette, A. Bücken, P. Krahwinkler, M. Emde: Realization of a highly accurate mobile robot system for multi purpose precision forestry applications, The 14th Int. Conf. Adv. Robotics (2009) pp. 133–138

56.51 J. Roßmann, P. Krahwinkler, A. Bücken: Mapping and navigation of mobile robots in natural environments. In: *Advances in Robotics Research – Theory, Implementation, Application. German Workshop on Robotics*, ed. by T. Kröger, F.M. Wahl (Springer, Berlin, Heidelberg 2009) pp. 43–52

56.52 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

56.53 P. Krahwinkler, J. Roßmann, B. Sondermann: Support vector machine based decision tree for very high resolution multispectral forest mapping, IEEE Int. Geosci. Remote Sens. Symp. (2011) pp. 43–46

56.54 A. Bücken, J. Roßmann: From the volumetric algorithm for single-tree delineation towards a fully-automated process for the generation of virtual forests. In: *Progress and New Trends in 3D Geoinformation Sciences. Lecture Notes in Geoinformation and Cartography*, ed. by J. Pouliot, S. Daniel, F. Hubert, A. Zamyadi (Springer, Berlin, Heidelberg 2013) pp. 79–99

56.55 J. Roßmann, T. Jung, M. Rast: Developing virtual testbeds for mobile robotic applications in the woods and on the moon, The IEEE/RSJ 2010 Int. Conf. Intell. Robots Syst. (2010) pp. 4952–4957

56.56 A. Shiriaev, L. Freidovich, I. Manchester, U. Mettin, P. La Hera, S. Westerberg: *Status of Smart Crane Lab Project: Modeling and Control for a Forwarder Crane*, Department of Applied Physics and Electronics (Umeå University, Umeå 2008)

56.57 U. Mettin, P.X. La Hera, D.O. Morales, A.S. Shiriaev, L.B. Freidovich, S. Westerberg: Trajectory planning and time-independent motion control for a kinematically redundant hydraulic manipulator, Int. Conf. Adv. Robotics (2009)

56.58 U. Mettin, S. Westerberg, P.X. La Hera, A. Shiriaev: Analysis of human-operated motions and trajectory replanning for kinematically redundant manipulators, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2009)

56.59 D. Morales, S. Westerberg, P. La Hera, U. Mettin, L. Freidovich, A. Shiriaev: Open-loop control experiments on driver assistance for crane forestry machines, IEEE Int. Conf. Robotics Autom. (2011)

56.60 A. Hansson, M. Servin: Semi-autonomous shared control of large-scale manipulator arms, Control Eng. Pract. **18**(9), 1069–1076 (2010)

56.61 S. Westerberg, I. Manchester, U. Mettin, P. La Hera, A. Shiriaev: Virtual environment teleoperation of a hydraulic forestry crane, IEEE Int. Conf. Robotics Autom. (2008)

56.62 Y.-C. Park, A. Shiriaev, S. Westerberg, S. Lee: 3D log recognition and pose estimation for robotic forestry machine, IEEE Int. Conf. Robotics Autom. (2011)

56.63 NLIS: National Livestock Identification System, NSW Department of Primary Industries, http://goo.gl/WoHUi2

56.64 Lely home page: http://goo.gl/JCl8Cl

56.65 Juno feed-pusher: http://goo.gl/V5H2Ye

56.66 Lely Vector feeder: http://goo.gl/dLiL3q

56.67 M. Dunn, J. Billingsley, N. Finch: Machine vision classification of animals, Mechatron. Mach. Vis. 2003: Future Trends: Proc. 10th Annual Conf. Mechatron. Mach. Vis. Pract. (Research Studies, Baldock 2003)

56.68 C. McCarthy, J. Billingsley, N. Finch, P. Murray, J. Gaughan: Cattle liveweight estimation using machine vision assessment of objective body measurements: First results, Proc. 28th Bienn. Aust. Soc. Anim. Production Conf., Vol. 28 (Australian Society of Animal Production, Wagga Wagga 2010)

56.69 Y. Wang, W. Yang, P. Winter, L.T. Walker: Non-contact sensing of hog weights by machine vision, Appl. Eng. Agric. **22**(4), 577 (2006)

56.70 W. Zhu, F. Zhong, X. Li: Automated monitoring system of pig behavior based on RFID and ARM-LINUX, Third Int. Symp. Intell. Inf. Technol. Secur. Inf. (2010) pp. 431–434

56.71 S. Barry, D. Cook, R. Duthie, D. Clifford, D. Anderson: Future Surveillance Needs for Honeybee Biosecurity, RIRDC Publication No. 10/107, http://goo.gl/MUrPLo

56.72 Lely Astronaut A4 milking system: http://goo.gl/wYhUVH

56.73 J.P. Trevelyan: Sensing and control for sheep-shearing robots, IEEE J. Robotics Autom. **5**(6), 716–727 (1989)

56.74 Robotics in the Poultry Industry, Poultry CRC: http://goo.gl/xH5vy6

56.75 A. J. Bubb, T. K. Driver, C. D. James: CASE STUDY - Redefining the cowboy: Precision pastoral decision making from remote monitoring and control of cattle, http://goo.gl/Vu0FIQ (2010)

56.76 R. Rankin: Further automation, Meat '93, Aust. Meat Ind. Res. Conf., Gold Coast. (1993), available online at http://goo.gl/Iv2yjh

56.77 Performance Audit, Management of Project Fututech, The Meat Research Corporation, http://goo.gl/C5jMgm

56.78 Robotic Technologies Limited: RTL – Scott and Silver Fern Farms, http://goo.gl/xZxL0Z

56.79 Automated Boning Room System, Scott Group: http://goo.gl/0zFVqi

56.80 Chicken Whole Leg Deboning Machine, Mayekawa: http://goo.gl/xhj2Lf

56.81 Poultry slaughtering, Hyfoma: http://goo.gl/dzTfVT

56.82 Association for Unmanned Vehicle Systems International: The Economic Impact of Unmanned Aircraft Systems Integration in the United States, http://goo.gl/NS5EkT (2013)

56.83 J. Dong, L. Carlone, G.C. Rains, T. Coolong, F. Dellaert: 4D mapping of fields using autonomous ground and aerial vehicles, ASABE Int. Conf., Montreal (2014), Paper No. 141912258

56.84 F.K. van Evert, P. van der Voet, E. van Valkengoed, L. Kooistra, C. Kempenaar: Satellite-based herbicide rate recommendation for potato haulm killing, Eur. J. Agron. **43**, 49–57 (2012)

56.85 C. Kempenaar, F.K. van Evert, T. Been: Use of vegetation indices in variable rate application of potato haulm killing herbicides, Proc. ICPA Conf., Sacramento, USA 2014 (2014)

56.86 C. Pugh, M. Jarman, S. Keyworth, J. Webber, I. Cameron: URSULA Agriculture, http://goo.gl/HzGkRT (2013)

56.87 S. Singh, S. Nuske, M. Bergerman, T. Bates, J. M. Peltier: Vineyard efficiency through high-resolution, spatiotemporal crop load measurement and management, http://goo.gl/iqXgWy

56.88 S. Blackmore: New concepts in agricultural automation, HGCA Conf. (2009)

56.89 J.L. Glancey, W.E. Kee: Engineering aspects of production and harvest mechanization for fresh and

Avidbots Ex1016
Page 1512 of 2154

processed vegetables, HortTechnology **15**, 76–79 (2005)
56.90 T. Burks, F. Villegas, M. Hannan, S. Flood, B. Sivaraman, V. Subramanian, J. Sikes: Engineering and horticultural aspects of robotic fruit harvesting: Opportunities and constraints, HortTechnology **15**, 79–87 (2005)
56.91 K. Ellis, T.A. Baugher, K. Lewis: Use of survey instruments to assess technology adoption for tree fruit production, HortTechnology **20**, 1043–1048 (2010)

Avidbots Ex1016
Page 1513 of 2154

Multimedia Contents



57. Robotics in Construction

Kamel S. Saidi, Thomas Bock, Christos Georgoulas

This chapter introduces various construction automation concepts that have been developed over the past few decades and presents examples of construction robots that are in current use (as of 2006) and/or in various stages of research and development. Section 57.1 presents an overview of the construction industry, which includes descriptions of the industry, the types of construction, and the typical construction project. The industry overview also discusses the concept of automation versus robotics in construction and breaks down the concept of robotics in construction into several levels of autonomy as well as other categories. Section 57.2 discusses some of the offsite applications of robotics in construction (such as for prefabrication), while Sect. 57.3 discusses the use of robots that perform a single task at the construction site. Section 57.4 introduces the concept of an integrated robotized construction site in which multiple robots/machines collaborate to build an entire structure. Section 57.5 discusses unsolved technical problems in construction robotics, which include interoperability, connection systems, tolerances, and power and communications. Finally, Sect. 57.6 discusses future directions in construction robotics and Sect. 57.7 gives some conclusions and suggests resources for further reading.

57.1 **Overview** 1494
57.1.1 Industry Description 1494
57.1.2 Automation in Construction 1496
57.1.3 Classification of Robotics in Construction 1496
57.2 **Offsite Applications of Robotics in Construction** 1499
57.2.1 Robotics in Component Production 1499
57.3 **Onsite Applications of Single Task Construction Robots** 1504
57.3.1 Steel Welding 1504
57.3.2 Reinforcement Manufacturing and Positioning 1504
57.3.3 Concrete Distribution 1504
57.3.4 Robots for Customized Construction On-Site 1505
57.3.5 Robotic Interior Finishing 1506
57.3.6 Robotic Facade Operation 1510
57.3.7 Earthmoving 1511
57.3.8 Road Maintenance 1511
57.3.9 Material Handling 1511
57.4 **Integrated Robotized Construction Sites** 1511
57.4.1 Robotic Roof Field Factory Approach 1511
57.4.2 Robotic Field Factory on-Ground Approach 1513
57.4.3 Robotic Deconstruction and Robotic Recycling 1513
57.5 **Currently Unsolved Technical Problems** . 1514
57.5.1 Interoperability 1514
57.5.2 Structural Connection Systems 1514
57.5.3 Tolerances 1515
57.5.4 Power and Communications in the Field 1515
57.5.5 Sensing 1516
57.6 **Future Directions** 1516
57.7 **Conclusions and Further Reading** 1516
Video-References 1517
References 1517

Construction is a ubiquitous human activity that relates to the creation or realization of physical artifacts or custom-made capital goods. It is distinguished from manufacturing in that the production activity normally occurs in a field setting and is undertaken in the open air, on natural terrain, and often with naturally oc-

Avidbots Ex1016
Page 1514 of 2154

curring materials. Typically, building and construction products are large in scale and unique in form. Moreover, the environment or field setting is typically unique and requires a, rather ad hoc, *factory* to be synthesized on site.

Over the centuries, various forms of machines and mechanical engineering systems have been introduced into the construction engineering domain and into the building and construction industry to increase production efficiency. In common with the fields of agriculture, mining, and forestry, the long-term trend has been for these fields to become increasingly mechanized [57.1].

In the last few decades, with the decrease in the relative cost of machinery to labor and with the globalization of markets, the construction industry has become significantly more capital intensive and large-scale machinery systems and pieces of construction plant – such as tunnel-boring machines and very large tower cranes – have become commonplace. This trend to mechanization is likely to continue with the progressive introduction of computer-controlled construction machinery and flexible manufacturing concepts into the industry.

With the relatively recent development of the microprocessor and the availability of low-cost computer and sensing technology, construction robots have become a technical and economic possibility and this form of technology is now gradually being used in the industry.

57.1 Overview

The application of robots in construction traditionally falls under *construction automation*.As the term implies, the field of construction automation is focused on automating construction processes, and the use of robots is but one aspect of automation. Construction processes also fall within several categories best described through a brief introduction to the construction industry.

57.1.1 Industry Description

The construction industry typically accounts for 5% of a country's gross domestic product (GDP) or gross value added (GVA) and employs a significant portion of the country's workforce. Table 57.1 presents some of these statistics for the USA, the European Union (EU), Japan and China [57.2–8]. Worldwide, construction industry *spending* was estimated at approximately 11% of the world's GDP at the end of the 20th century [57.9].

Construction is considered by many to be technologically behind other industries, such as manufacturing. In manufacturing, a product is designed for mass production, whereas construction *products* (or projects) are usually one-off and unique [57.10]. Thus the efficiencies achieved through mass production are not easily achieved in construction.

Table 57.1 Value of construction put in place and number of construction workers for the USA, EU, Japan and China

Country Data period (year)	USA 2012	EU 2009	Japan 2011	China 2011
Value of construction put in place as percentage of national GDP/GVA (%)	5	7	10	7
Number of construction workers as percentage of the total national workforce (%)	5	11	10	5
Total number of workers in the national workforce (million)	112	175	63	784

Other often attributed reasons for the construction industry's technological lag are the industry's fragmentation and aversion to the risks associated with the introduction of new technologies [57.10, 11]. For example, in the US in 2010 firms with fewer than twenty employees employed approximately 40% of all construction workers, and 63% of all construction workers worked for specialty trade contractors who accounted for 64% of construction firms [57.10]. Specialty trade contractors are usually subcontractors on a project and are not responsible for the overall outcome of the project.

In addition, unlike their manufacturing counterparts, construction sites are for the most part unstructured, cluttered, and congested, making them difficult environments for robots to operate in. Furthermore, human workers are also present in large numbers on a construction project, making safety a paramount concern.

Types of Construction

Construction projects are usually classified as residential, commercial, industrial, or civil. Residential construction generally involves single-family homes or large apartment buildings; commercial focuses on building structures such as office and retail space, warehouses, and so on; industrial is involved in building factories, power plants, and other similar structures; and civil construction focuses on public infrastructure such as highways, bridges, tunnels, and dams.

The Typical Construction Project

A construction project typically goes through six major phases, as shown in Fig. 57.1. Some projects may go through different variations of the sequence shown, but most projects include planning, design, construction, and operation phases [57.4, 10]. Phase 1 begins when a need for a project arises and the requirements are identified. Phase 2 involves developing alternative project plans that could meet the identified needs and evaluating the technological and economic feasibility of each alternative. Phase 3 develops detailed engineering designs and specifications for the plan selected in phase 2. The construction of the facility from ground-breaking through to final inspection takes place in phase 4 of the project. The facility is occupied and commences operation in phase 5 and continues operating until it is time to shut down and dismantle the facility once it becomes obsolete (phase 6).

The actual physical work (building, operating, maintaining, and dismantling) on a construction project takes place in the last three phases. Although these are the only phases in which machines are used, construction automation can also take place during the other phases of a project, as described in Sect. 57.1.2.

During the various phases of a construction project, several stakeholders may be involved at any time. The major stakeholders include the following: the *owner* and *operator*, whose needs initiated the project; the *architect* and *engineer*, who have the task of translating the owner's needs into an aesthetically pleasing and structurally sound design; and the *general contractor*, whose task is to translate the design into a physical structure. In addition several other stakeholders may be involved either independently from the above stakeholders or as part of their organizations; for example, the constructor may employ a project manager, a construction manager, a site superintendent, and others, and he/she will typically subcontract major activities such as excavation and concrete pouring. The subcontractors and sub-stakeholders are often the ones who use a new technology and can either *make or break* its implementation.

Although the builder is the most likely user of robots on a construction project, the actual work on site is often conducted by subcontractors who often are reluctant or financially unable to use advanced technologies which have not been entirely adopted by industry. A constructor's goal is often to meet the owner's requirements by the most efficient *and* least risky methods possible. Hence, traditional construction methods that have stood the test of time are preferred. Nevertheless, it has often been anecdotally reported that the owner has *the power* to require the use of certain technologies on a construction project, since that shifts part of the risk and cost associated with the use of the technology to the owner. This phenomenon may be partially responsible for the widespread penetration of laser scanning technology into the construction industry that is currently taking place [57.12].

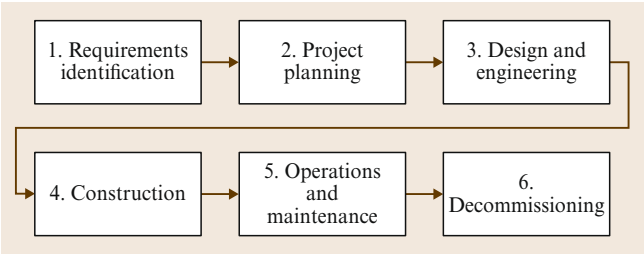


Fig. 57.1 Phases of a typical construction project

Typical Construction Processes

From a purely physical-world and practical point of view, construction may be viewed as being comprised of a finite number of elementary processes [57.13–16], which may be summarized by the following list [57.11]:

- Attaching
- Building
- Coating
- Concreting
- Connecting
- Covering
- Cutting
- Digging
- Finishing
- Inlaying
- Inspecting
- Jointing
- Measuring
- Placing
- Planning
- Positioning
- Spraying
- Spreading.

Most of these processes can also be grouped into three predominant types of functional operators as follows:

- Materials handling (by bulk and unit load)
- Materials shaping (cutting, breaking, compacting, and machining)
- Structural joining.

These functional operators are typically each applied to multiple operands. Common operands in build-

ing and civil engineering are steel and other metals, concrete, timber, earth and rock, masonry, plastic and glass, cement, aggregate, epoxy resin, bitumen, and other bulk and formed materials.

57.1.2 Automation in Construction

Construction automation describes the field of research and development focused on automating construction processes, and the use of robots is but one aspect of that field. In short, construction automation deals with applying the principles of industrial automation to the construction sector, whether in building construction, civil engineering (roadways, dams, bridges, etc.), or in prefabrication of construction components [57.17]. This can be viewed as an extension to research in field service robots generally designed to replace or assist humans in a specific construction-related task or function.

From a historical perspective, research in construction robotics and automation started in the 1980s with the introduction of single-purpose robots (principally remotely controlled, or teleoperated, machines). The Japanese led this effort, driven primarily out of concern for societal demographics, which showed a significant future shortfall in personnel available for the construction labor pool [57.18]. In the US the principal related work involved developing remote control or teleoperated machinery for hazardous work that required modified construction equipment. Example applications include robots developed for rapid runway repair and unexploded ordinance removal. In the EU, research was focused on the development of large-size masonry (brick laying, assembly) robots for residential and industrial building construction.

During the next decade, as research on task-specific construction robots continued, large Japanese construction firms introduced on-site *factories* for high-rise construction. These construction systems included just-in-time delivery of components, automated part tracking and material handling, robotic connection and assembly, and centralized control in an enclosed or semi-enclosed environment. It is reported that the systems enable better working conditions (weather invariant) and reduced project completion time [57.19]. Other benefits include improved productivity and quality, though overall construction costs are not necessarily lower [57.20].

Advanced concepts in integrated residential construction automation were developed as part of the EU FutureHome project. In this construction concept, each structure consists of several high-quality, prefabricated three-dimensional (3-D) modules and two-dimensional (2-D) panels which are assembled *production-style* on site [57.21]. An analogue of this approach for automated residential construction has been commercialized in Japan [57.22].

New methods for collecting, processing, analyzing, and communicating construction information are a significant area of construction automation research [57.23]. This research includes data interoperability and exchange through the design, construction, operations, maintenance, and decommission phases of a capital project [57.24]; advanced sensors for assessing the status of the construction process [57.25–27]; visualization systems for planning construction events, verifying constructability, and maintaining site situational awareness [57.28–30]; and information models which extend traditional computer-aided design (CAD) modeling to combine both the physical (geometric) and functional characteristics of building components [57.31].

Finally, three notable, large-scale, EU projects attempted to integrate construction automation into the way construction projects are executed: the ManuBuild (open building manufacturing) project dealt with, among other things, the development of mobile field factories, including robots, for on-site modular construction [57.32]; the I3CON (industrialized, integrated, intelligent, construction) project dealt with the indoor automation and robotization of buildings [57.33]; and the Tunconstruct project dealt with the robotization of inspection and maintenance operations in tunnels [57.34].

57.1.3 Classification of Robotics in Construction

Construction robotics is an advanced form of mechanization (automation) in which an endeavour is made to automate some industrially important operation and thereby reduce the cost of this operation by either removing a human operator *from the control loop*, or enhance operational efficiency through machine control systems. Due to the nature of construction work, most robots which have been developed for the construction industry are either mobile or relocatable systems. Some platforms, such as floor-finishing robots and machine-controlled earthmovers require mobility as a specific function of the work process to be performed. Others, such as wall and ceiling panel manipulators, require some level of mobility to extend their operating workspace.

Robot terminology can vary depending upon the research discipline. For this chapter two broad classifications of construction robotics are onsite robots and offsite robots. These are distinguished by whether they are applied at the construction site or at a factory or prefabrication facility. In addition, a further distinction can

be made depending on whether a construction robot is intended to be used for a single task (onsite or offsite) or whether multiple robots are integrated into an automated construction site.

Single Task Construction Robots Versus Integrated Robotized Construction Sites

After the initial experiments in large-scale industrialized, automated and robotized pre-fabrication of system houses were conducted successfully in Japan, and the first products (e.g. Sekisui M1) also proved successful in the market, in 1975 a Japanese construction contractor, Shimizu Corporation in Tokyo, set up a research group for construction robots. The goal was now no longer the mere shifting of complexity into a structured environment as in prefabrication, but the development and deployment of systems that could be used on the construction site to create structures and buildings. The focus initially was set on simple systems in the form of so-called single task construction robots that can execute a single, specific construction task in a repetitive manner.

Single-task construction robots are systems that support workers in executing one specific construction process or task (e.g. digging, concrete levelling, concrete finishing, painting) or completely supplement the physical activity of the human workers necessary to perform this one process or task. Further, the processes and tasks they support or supplement can be allocated to a specific profession or craft. In addition, the processes and tasks for which single task robots were developed had in common that they entailed frequent, repetitive activities. Common characteristics of single-task construction robots are as follows:

- Highly specialized not only for a profession but even for a task within a profession (e.g., concrete pouring, levelling and finishing.
- Enhanced productivity compared to conventional labour. E.g., according to [57.17] the conventional labour productivity rate for concrete floor finishing is between $100\,m^3/h$ and $120\,m^3/h$, whereas the productivity using machines is between $300\,m^3/h$ and $800\,m^3/h$.
- Improved quality through precise control of functions and operations (e.g., uniform distribution of paint) and by allowing real time monitoring (and recording) of the operation.
- Improved working conditions by removing workers from dangerous environments and reducing the amount of and heavy physical work.
- Most robots allow various operation modes such as autonomous or sensor guided, pre-programmed, or tele-operated.
- Reduced material consumption through precise control of material delivery and collection and reuse of unused material.
- Most robots have simple yet robust sensor technologies, such as gyroscopes, simple laser systems, or touch/pressure sensors.
- Most robots only require 1 or 2 persons to operate.

The evolution of industrialized and automated building prefabrication during the 1970s along with the development of single tasks robots, envisioned the concept of integrated robotized construction sites. This concept combined prefabrication technology (processing of prefabricated components instead of parts in order to reduce complexity on the site), single task construction robotic systems, and moving or stationary site factories that were able to assemble the building's main structure (e.g., steel frame or concrete structure) almost automatically. As those site factories not only automated parts of the construction process but also integrated prefabricated component technology and single task automation can thus be called as integrated robotized construction sites.

A major reason for the transition into integrated robotized construction sites was that the construction companies realized that single task construction robots which were not networked or embedded within a greater infrastructure, turned out to be incompatible with the way buildings were designed and built. Single task robots were designed to execute certain tasks meanwhile construction workers were not allowed to interfere significantly with the robots activity. However, it turned out that under these premises, only a very few number of robots could be used efficiently. The constraints for the workers, the necessary safety regulations, coupled with the unforeseen, unpredictable, and dynamic processes at the construction site, led to the implementation of individual robots working in parallel. Although single task construction robots achieved a high throughput, significant time had to be spent on-site for transportation, preparation, programming, configuration, etc. Besides the mentioned frictions single task construction robots caused on (conventional) construction sites, the emergence of concepts for integrated sites was also nurtured by the new technological possibilities. Important for the integration of such systems into larger and coordinated automated systems was the development of systems that allowed controlling and monitoring an uninterrupted flow of information and material on-site, between individual automated entities that are involved in the final assembly of the building.

The transition to integrated sites can also be explained from an evolutionary view. In many industries

the evolution from more workshop like production with individual and only loosely coupled production entities or stations, to flow-line like or production line like systems with stable processes and continuous material flow was part of the evolution of other well established and known industries,, i. e., automotive, computer industry.

Finally, three general categories of construction robots are introduced. The first class is *teleoperated systems*, which for simplicity includes *remote control systems*. The general distinction between the two terms is whether or not the equipment must be operated in line of sight from the human controller [57.35]. The second category, *programmable construction machines* (PCM), includes most construction equipment that is outfitted with sensors and mechanisms to augment operation by an onboard human operator. The final category, *intelligent systems*, relates to unmanned construction robots which operate either in a semi- or fully autonomous mode. In Sect. 57.3 this classification will be expanded to include examples based on various generic activities, materials handled (operand types), levels of onboard intelligence, levels of commercialization, and levels of system integration and computer integration.

Teleoperated Systems in Construction

In established engineering terminology, the term *teleoperation* refers to the remote control of machines and systems. In teleoperation (loosely referred to as telerobotics) the control of the machine is accomplished by the use of remote control means such an umbilical cord or wireless control. Teleoperation ideas and methods are used extensively in the space and nuclear industries (Chaps. 55 and 58, respectively).

In telerobotics, the machine does not operate autonomously but is under the control of a human. Data sensing and interpretation and cognitive activities such as task planning are done by the operator.

Recently, many telerobotic devices have appeared in the construction and mining industries. These machines have evolved in response to industrial situations where there is danger to the operator and where remote-controlled machinery is necessary (e.g., teleoperated small compactors). Situations of this kind occur in the construction, demolition, and mining industries and in other hazardous locations.

The technology for telerobotics in construction is well established, with a number of excellent examples of such activity available, for example, a sophisticated model-based supervisory-control-type distributed teleoperation system for the construction of a trench for a diversion dam in a lava field in Japan using a fleet of heavy earthmoving machines [57.36].

Programmable Construction Machines

A software-programmable construction machine is what most people would consider to be a robot. The operator of this type of machine is able to vary the task to be accomplished within certain constraints either by choosing from a preprogrammed menu of functions or by *teaching* the machine a new function. Variations in the task to be accomplished could be as simple as slight changes in the driving speed based on the current load for an automated forklift, or as complex as a change from being able to pick and place steel beams and columns using an autonomous crane to being able to deliver concrete autonomously using the same crane.

Generally, software-programmable construction machines are identical to traditional construction machines (such as an excavator), but have been modified to be controllable through a computer (similar to the way in which traditional manufacturing machines – such as mills and lathes – evolved into computer numerically controlled (CNC) machines).

Software-programmable construction machines can make use of an electronic representation of a portion of the construction site where their work is to be conducted in order to control all or part of the machine's operation. A commercialized example is *stakeless grading*, where data from a 3-D model is used in combination with global positioning systems (GPS) and/or laser measurement systems to automate the blade control for bulldozers and motor-graders.

Intelligent Systems in Construction

As opposed to a teleoperated or a software-programmable construction machine a fully autonomous construction robot is expected to accomplish its task, within a defined scope, without human intervention. A semiautonomous construction robot would be expected to accomplish its task with some level of planning interaction conducted with a human supervisor [57.35]. In each case the construction robot is expected to adapt to its sensed environment, formulate plans for the execution of its task, and replan as necessary (with possibly some human assistance in the semiautonomous mode). The intelligent construction robot should also be able to determine when its tasking is not executable and request assistance.

Example research in intelligent construction systems includes autonomous excavation [57.37, 38] and autonomous crane operations [57.30, 39–41].

57.2 Offsite Applications of Robotics in Construction

The infusion of technologies (such as computer aided design (CAD), computer aided manufacturing (CAM)) into the construction industry from other industries (such as the automotive, airplane and shipbuilding industries) are the clearest proof for a shift in building strategy. Fully automated concrete fabrication is currently a reality that are implemented by factories with customized product delivery in order to adapt to the changing requirements of the market. There are two major industrialized production methods for prefabricated concrete: Production methods that (a) use stationary single formwork and (b) that use mobile formwork.

57.2.1 Robotics in Component Production

Production methods using stationary single formwork comprise mobile workflows such as cleaning, pouring/casting and moving the final product to the storage, while the formwork is stationary during the whole process of prefabrication. In production methods using mobile formwork, the work posts are stationary while the formwork is mobile, moving into the various prefabrication posts of the assembly line.

Concrete Production

A plant for combined manufacture of 2-D and 3-D precast concrete elements includes overhead gantry cranes and electric hoists, used to transport consumable materials (such as lattice girders, steel coils, and parts to be inserted/installed) as well as to lift elements from the concreting station into the storage facility or onto transport pallets. Pallets start at the discharging station, from where they proceed via the pallet cleaning station to the first workstation. There, by means of a de-palletizing device attached to a panel-stacking crane, the finished flooring elements are lifted off the pallets and stacked directly within the working range of the discharging vehicle. A cleaning and plotting (magazining, cleaning, plotting (MCP)) robot is used for a variety of tasks: picking up, insertion of latitudinal anchors, cleaning of pallets, full scale plotting of elements, and installation of latitudinal anchors (Fig. 57.2). Figure 57.3 shows further examples of robotic concrete component manufacturing and handling.

Brickwork Component Production

Since the late 1970s, due to the lack of skilled workers in construction and increasing costs of buildings, rationalization developments were started. They mainly took place in the masonry and the formwork sector, where no ready solutions from other countries were available. In Germany and also in most parts of Europe an often used and still loved building material is brickwork. Most of the building projects are constructed with this material. Due to the lack of brickwork in Japan (not resistant to earthquakes if not reinforced), and missing developments in other countries, main efforts were made in this field.



Fig. 57.2 (a) Pallet in concreting station; (b) bridge crane; (c) pallets molding systems; (d) demolding-depalletizing device; (e) cleaning, measuring and oiling (after [57.42])

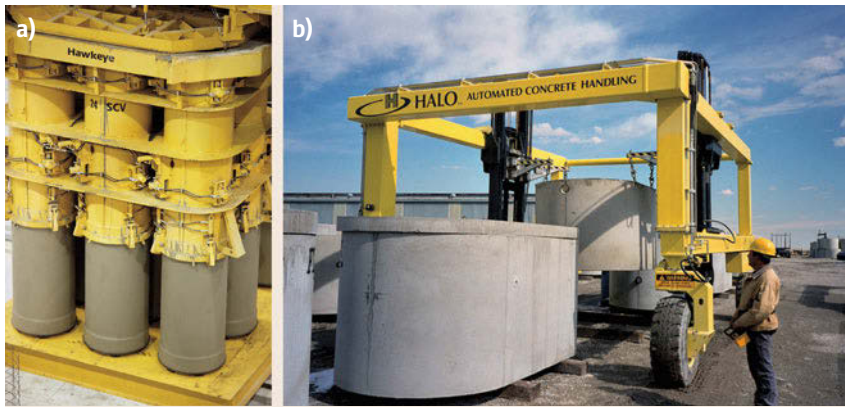


Fig. 57.3 **(a)** Automated concrete pipe manufacturing machines (courtesy Hawkeye Group USA); **(b)** precast concrete handling robot (courtesy Halo, Inc.)

In the last few decades activities to develop machines that support the laying of bricks and even automated robots have been increasing, especially in Germany. Brick laying is a very labor-intensive activity, which can lead to significant health issues and early retirement [57.43, 44]. Consequently a very small number of skilled workers is available nowadays. This in turn is reflected in ultimately high prices and wages, which will make brick building in some time prohibitively expensive. Therefore it should just be a question of time before only houses are put on the market that have not been erected on site by a team of bricklayers, but by a robot in the production shop.

Early Trials of Robotic Assembly of Modular Blocks

One idea for Robot Oriented Design (ROD) [57.45–47], in building construction is to change the conventional construction works adaptable to robotics so that construction system and advanced automation and/or robotics are co-adapted. Thus a structural system for the wall erection named SMAS (Solid Material Assembly System) [57.48] was proposed and developed already in the 80s in Japan. SMAS is a of reinforced masonry construction system. A standard building component of this system, 30 cm × 30 cm × 18 cm in size and 20 kg in weight, is made of pre-cast concrete and includes cross-shaped steel bar inside each component for the reinforced of structural wall. Components are positioned automatically by the robot one by one without arrangements of conventional bonding. Following the positioning of each components, steel bars are connected to those of adjacent components also by the robot. The joint type of steel bar for vertical direction is mechanical, and that for lateral direction is overlapping. Concrete is grouted from the top of the wall which is erected one storey high (about 3 m). The newly developed operating hand is installed to the mother robot (6-articulation-type robot) which was developed for a wide variety of applications in factory use and a series of experiments for wall erection were carried out.

The rapid progress being achieved today in the modernization and industrialization of building construction technology has triggered a trend to reduce the complicate works at construction site and increasingly produce building components at factories. It is obviously that the prefabrication has been successful in up-grading the quality of the building and in shortening the construction period. The sizes of those building components such as prefabricated structural members are also becoming larger to simplify assembling work at the construction site.

However, these movements are not necessarily oriented toward the introduction of robots. The heavy and large components are difficult to be operated by the robots and the complicated assemble techniques are sometimes too skilful for robots. Meanwhile, when one looks at the prefabrication of compact and lightweight structural components as a means of accomplishing construction work more efficiently, robots can be used for assembling these structural members efficiently.

From this point of view SMAS structural system was developed and proposed as a robot-oriented construction system. A masonry structure has not been considered as a major structural system in Japan because of earthquakes; however, it becomes to be recognized as a flexible structural system applied to various building designs when properly reinforced. In addition, it has advantages in the construction cost and in the construction period. The SMAS system itself was designed as a modular system for the purpose that single components could be developed further in the future. All system components including stones are developed as complementary parts according to the guidelines of Robot Oriented Design (ROD) (Fig. 57.4). These robotic approaches were meant to be applied on-site, but technologically they were proven to be not effective. Thus, they became better suited for offsite assembly operations.

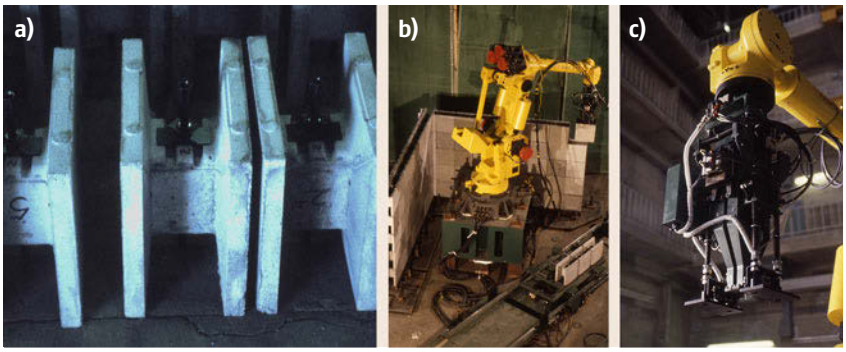


Fig. 57.4 **(a)** Building component developed by guidelines of Robot Oriented Design (ROD), **(b)** SMAS total system, **(c)** robot end-effector/gripper with optical fiber sensor guiding screwing mechanism

Automated Brickwork Plants

The adoption of CAD/CAM systems combined with implementation of Enterprise Resource Planning (ERP) Solutions has made the implementation of advanced technically refined, fully automatic production plants possible:

- High degree of information and communications technology (ICT) integrated and automated/interconnected processes
- Integration of as many devices as possible by ICT (interoperability)
- CAD/CAM Systems
- Integration of processes by ERP Systems
- Just in Time and Just in Sequence supporting customization.

Brickwork Robot Plant SüBA

Crucial technical developments preceded the construction of a fully automatic brickwork machine. The prototype of the automatic brickwork plant was developed by the company SüBA in Hockenheim, and the company Windhoff AG in Rheine at the beginning of the 1990s. The production of brickwork panes is appropriate for a capacity of 300 m^2 net area of brickwork panes – without windows and door recesses – in a shift of eight hours. The employment of CAD in the architect's offices made it possible to transfer the large data set for the production of brickwork panes directly without manual input over CAM to the brickwork robots. Considering the large number of data for the automat, which was necessary for the production of a brickwork pane, this was one of the most important problems of the automation of bricking at that time. Figures 57.5 and 57.6 give an overview over the most important SüBA factory modules and automation processes.

Brickwork Robot Plant Winkelmann (Horizontal Brickwork Panel Production)

Today's fully automated and highly robotized brickwork plants can be distinguished into two basic types: horizontal and vertical brickwork panel production. The brickwork robot plant Winkelmann is a characteristic example for horizontal panel production. All single devices are equipped with Microsystems, interconnected and part of a systemic logistic network. CAD/CAM guarantees efficient data processing between planning section and production. After the delivery standard palettes bricks are brought into processing order by an automated palletizing system with a robot for distribution of bricks (Fig. 57.7a). After that the bricks are taken up by automated de-palletizing system (Fig. 57.7b), which supplies horizontal brickwork layering robot station (Fig. 57.8a). Insertion of reinforcement and house

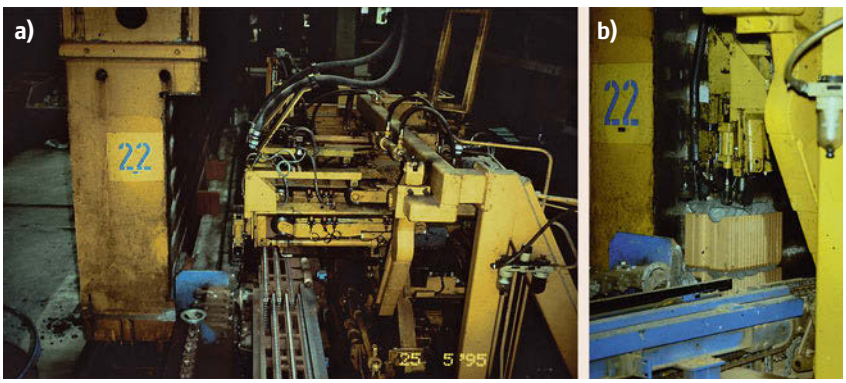


Fig. 57.5 **(a)** Automated reinforcement (horizontal) station; **(b)** mortar dispensing

Avidbots Ex1016
Page 1522 of 2154

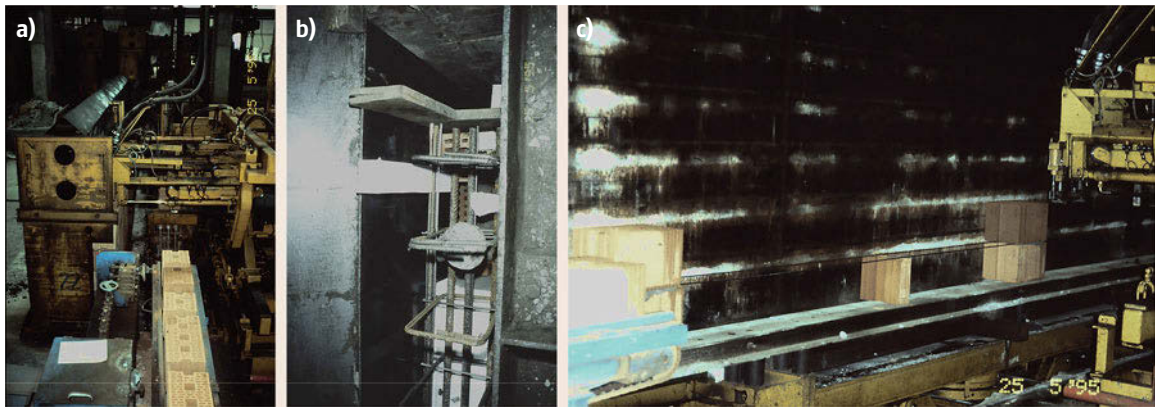


Fig. 57.6 (a) Brickwork positioning; (b) brickwork alignment; (c) automated reinforcement (vertical) station

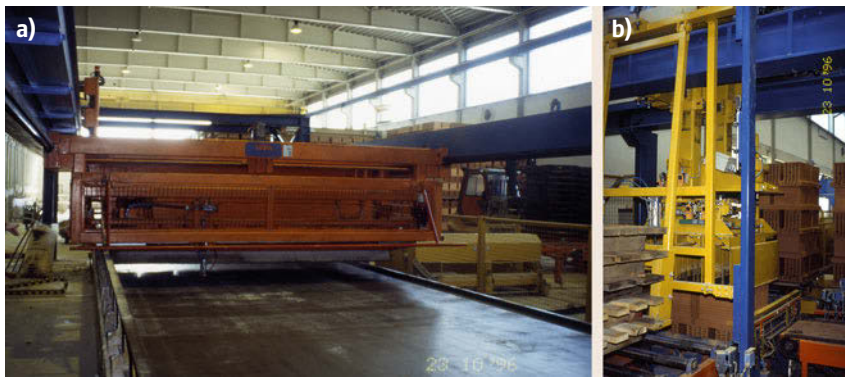


Fig. 57.7 (a) Automated palletizing system with robot for distribution of bricks; (b) automated de-palletizing system



Fig. 57.8 (a) Automated horizontal brickwork layering robot station; (b) insertion of reinforcement and house infrastructure

infrastructure as well as plastering are done stationary in the factory (Fig. 57.8b) until a finalized and dried-out brickwork panel is delivered to the construction site.

Brickwork Robot Plant Leonhard Weiss (Vertical Brickwork Panel Production)

The brickwork robot plant Leonhard Weiss is a characteristic example for vertical brickwork panel production. Many processes as for example the palletizing and de-palletizing to bring the bricks into factory order are similar to the horizontal type. Yet the central process of positioning the bricks in given order is done vertically layer by layer. High accuracy robots combined with linear axis are here in charge for the exact positioning. The automated vertical brickwork layering has some advantages in terms of efficient use of the factory area and moreover the firmness of the wall could be improved whereas the exact positioning of reinforcement, cables and other elements of the house infrastructure is done easier with the horizontal type. CAD/CAM combined

Avidbots Ex1016
Page 1523 of 2154

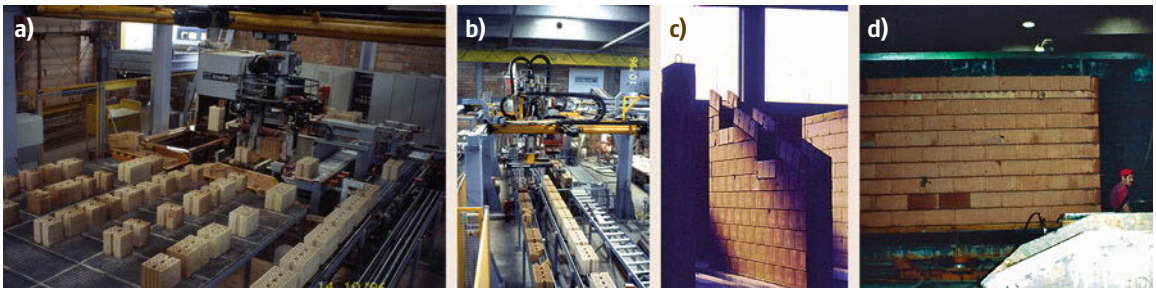


Fig. 57.9 **(a)** Sequencing station; **(b)** robotic assembly system for vertical brickwork layering; **(c,d)** final brickwork panel product

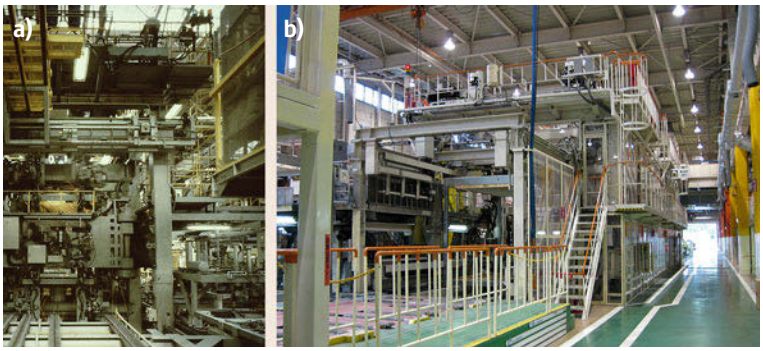


Fig.57.10a,b 3-D adjustable automated assembly welding unit (Sekisui Heim)

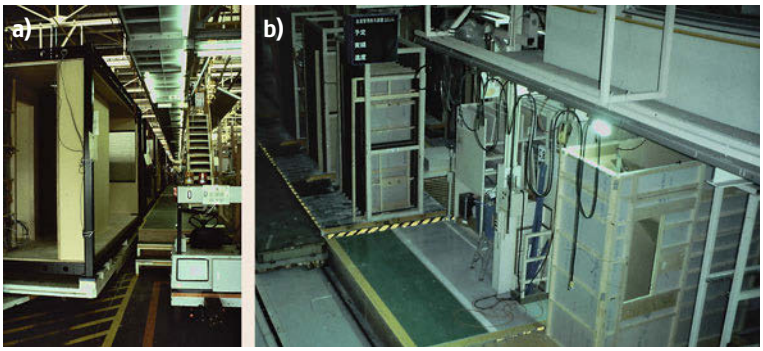


Fig. 57.11 **(a)** On the 400 m production-line, the steel frame units (*chassis*) pass several workstations; **(b)** 10 Insertion of prepared elements and by suppliers (e.g., Toto, Inax) prefabricated bath modules into the chassis unit

with variable production and ICT integrated production allows individually fabricated brickwork panels and necessary variations. Figure 57.9 shows the various steps of the automated vertical brickwork panel construction process.

Steel Component Production

Sekisui Heim introduced an automated steel frame production. One of the basic features is the automated assembling and welding station. Ceiling elements, flooring elements and columns are fed into this station, followed by automatic welding into a frame, which is used as chassis and bearing structure during the further completion process on the production line (Fig. 57.10). After the automated welding process, the steel frame chassis is streaming through the factory from work step to work step, until all installations have been completed (Fig. 57.11a). The factories of Sekisui and Toyota have gates on both sides of the assembly lines in order to receive material, parts, components and prefabricated bath or kitchen modules, required for the customized production of individual units. All of them arrive just-in-time and just-in-sequence by cooperating suppliers (Fig. 57.11b).

Avidbots Ex1016
Page 1524 of 2154

57.3 Onsite Applications of Single Task Construction Robots

The construction of larger buildings that use steel as the structural support system can involve a high total amount of welds. If the design of the columns and beams can be adjusted in order to reduce the amount and variety of welds, welding becomes a highly repetitive operation suitable for being automated.

57.3.1 Steel Welding

Automated welding is able to control and guarantee the quality of the connection between the welded parts to a level similar to (and sometimes better than) that achieved by professional human welders. Manual welding of these and other types of connections requires highly specialized skills that are currently in short supply. Furthermore, conventional welding can have damaging long-term effects on a worker's vision. Simultaneous automated welding on a beam (e.g., from two or three coordinated positions) is even able to ensure that the steel component is not distorted by the welding operation itself and thus improves the quality of the constructed structures. In Fig. 57.12a the Shimizu steel-welding robot is depicted. The robot can automatically weld a column including the corner portions. The configuration of a joint is detected by a laser sensor, and the welding is performed in an optimized way referring to a database. In Fig. 57.12b, the Obayashi Corporation Steel welding robot operates in a similar principle, using a circular clamp-on type fastener to the column. The welding process is computer-controlled, though workers are still involved in overseeing operations at least for the time being.

57.3.2 Reinforcement Manufacturing and Positioning

Concrete reinforcement operations involve cutting and bending of rebar (reinforcement bars), precise (relative to each other) arrangement of those rebar, binding of rebar and final positioning of the rebar elements or mesh on a floor or in a mold or formwork system. These tasks are labor intensive and hard on a worker's body (e.g., back accidents and damage to a worker's musculoskeletal system occur about ten times more during reinforcement work than during painting work [57.49]). Automated systems mitigate the risks and impacts on the health of workers and enhance the quality of the reinforced concrete structures. Figure 57.13a shows a reinforcement production robot and Fig. 57.13b shows a teleoperated rebar placing robot.

57.3.3 Concrete Distribution

Concrete distribution systems are used to distribute mixed concrete with uniform quality over large surfaces or over formwork systems. Concrete distribution

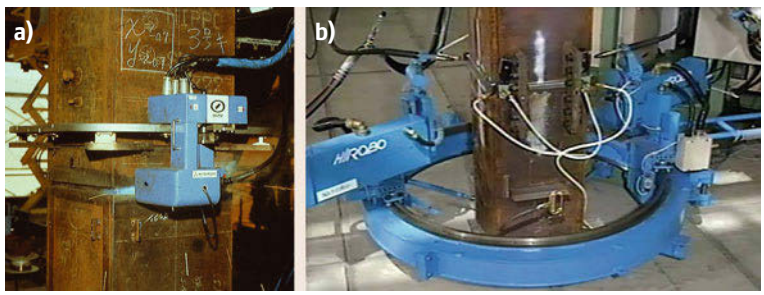


Fig.57.12a,b Steel welding robots. **(a)** Shimizu Corp.; **(b)** Obayashi Corp

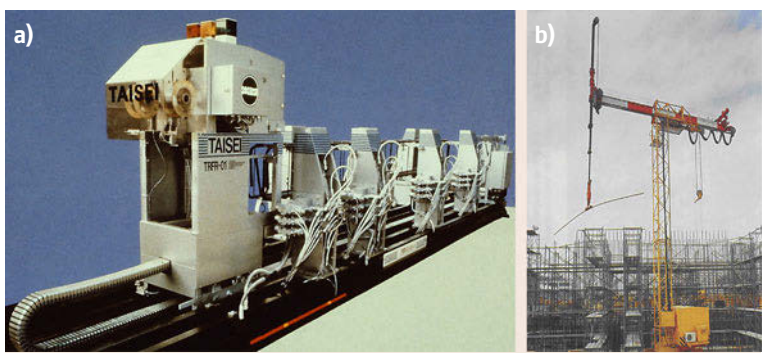


Fig. 57.13 **(a)** Automated reinforcement production and **(b)** teleoperated rebar placement (after [57.17])

Avidbots Ex1016
Page 1525 of 2154

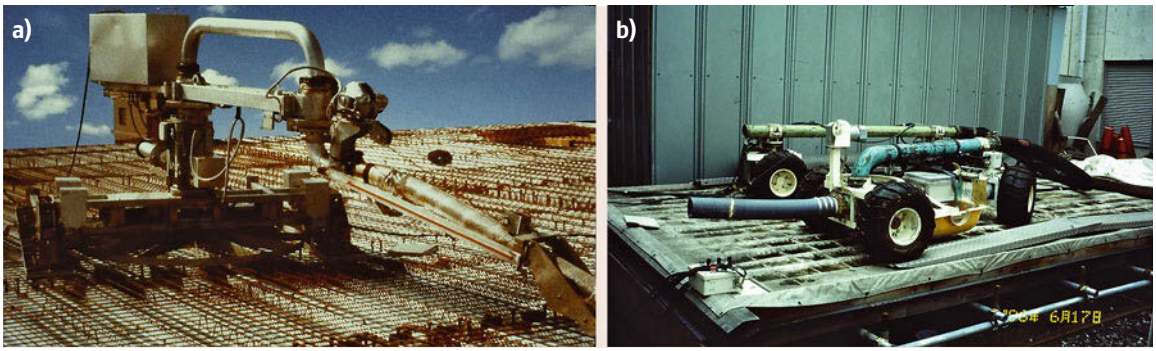


Fig. 57.14 (a) Takenaka Corp. Automatic Concrete Distributor *DB Robo*; (b) Tokyu Concrete Distribution Robots



Fig. 57.15 Concrete slipform road paving machine (courtesy GOMACO Corp.) ◄

covers the continuous supply with concrete (pumps, hoses), a system which slides in a certain pattern over the area where the concrete has to be distributed and a concrete ejection system. Systems can be operated manually, tele-operated and sometimes sensor-guided or fully automatic. Depending on the surface where the concrete needs to be distributed, concrete distribution systems can be truck mounted, stationary or mobile. In Fig. 57.14a, a track mounted automatic concrete distributor robot (DB Robo) can be seen. In Fig. 57.14b a mobile concrete distributor system can be seen, which due to its wheel-based platform, enables an enhanced operating range. Figures 57.15 and 57.16 show other examples of automated concrete distribution systems.

57.3.4 Robots for Customized Construction On-Site

Several masonry-wall erecting robots have been developed to-date. Examples of these are the SMAS in Japan

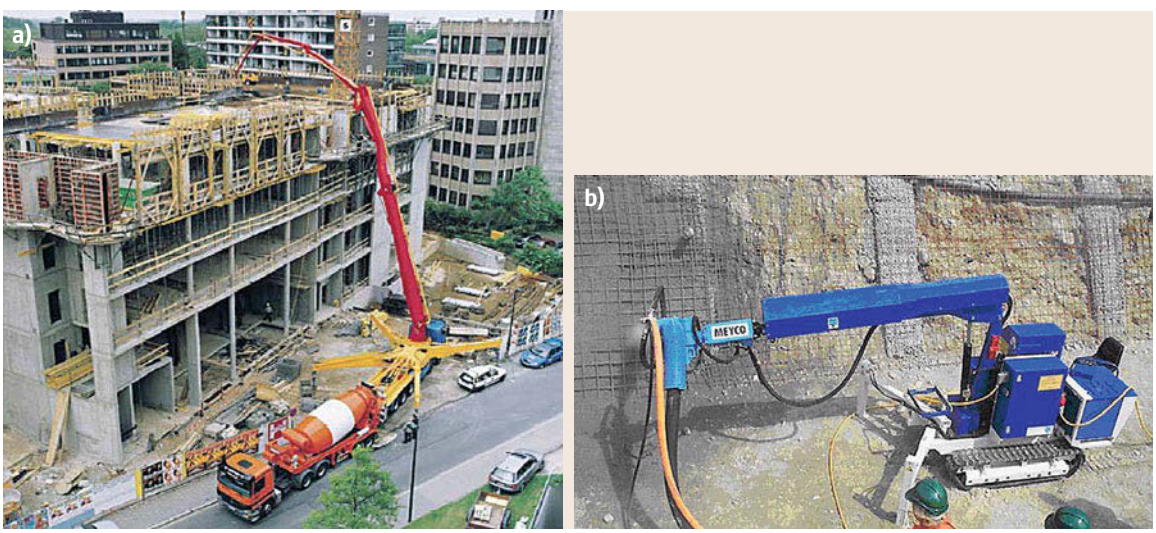


Fig. 57.16 (a) Programmable, articulated-boom machine for the pumped delivery of fresh concrete (after [57.50]) and (b) teleoperated concrete spraying robot (courtesy MEYCO Equipment)

Avidbots Ex1016
Page 1526 of 2154

and the ROCCO robot at the University of Stuttgart. Many of these systems are still in the prototype stage. The basic concepts that these systems have in common are as follows:

- Autonomous mobility on the construction site
- Sensor systems determine the robot's position and orientation in its environment
- Automatic pick up of the bricks from the pallets
- Automatic application of mortar
- Automatic positioning of the bricks.

ROCCO is a Robot Construction System [57.51, 52], for Computer Integrated Construction (CIC). Several companies and institutes participated in this EU funded project in an inter-disciplinary and international approach with experts in the fields of construction technology, mechanical and electrical engineering and information technology from Germany, Spain and Belgium. The goal of the project was the development of a computer integrated robot system, which also contains a continuous solution in the ICT (Information Communication Technology) for all steps from the architectural design to the automated assembly of the components on the construction site.

The main emphasis of the project lies on the realization of a mobile robot system for construction site operation as well as on the integration of a computer based system for work preparation and quality control. In the work preparation phase the necessary data is generated for the pre-fabrication and customization of the masonry blocks, for the construction site layout and for the automatic robot program generation.

Based on the CAD-representation of the building, first the walls are divided automatically into the necessary blocks. In the next step the optimal working positions of the mobile robot as well as the positions of the pallets and the arrangement of the blocks on the pallets are automatically calculated. With this information the necessary non-standard and standard blocks can be produced and palletized. Finally the robot programs are generated automatically out of the calculated geometry information. The user interface on the construction site is graphically interactive and enables the user to partially re-program the generated robot programs to deal with the uncertainties of the construction process without the need to learn a specific robot programming language.

To test different approaches based on the construction application (residential buildings vs. industrial) and the sensor integration (autonomous vehicle vs. long reach), two systems were developed within the project. The first one, for residential buildings, has a reach of 4.5 m and it is able to handle up to 400 kg. This robot is placed over an autonomous vehicle that allows the movement on the construction site, Fig. 57.17a. Its main task is the erection of walls in residential buildings. The second one is able to handle up to 1000 kg with a reach of 8.5 m. This robot, Fig. 57.17b, is placed over a towable platform and its main task is the erection of external walls in industrial buildings with typical height of up to 8 m and standardized layout.

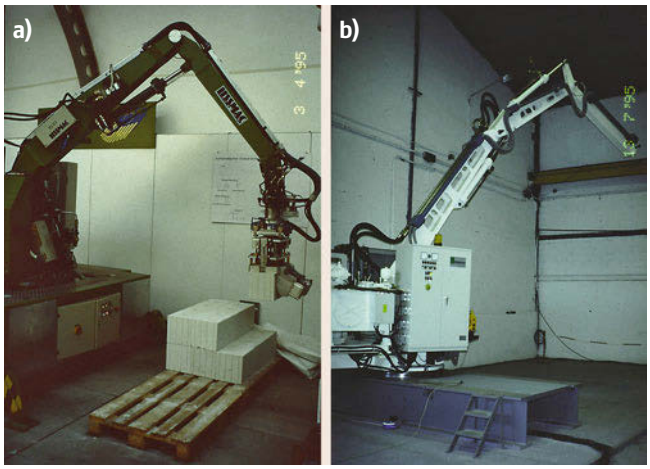


Fig. 57.17 (a) Robot for residential buildings; (b) robot for industrial use with long reach manipulator

57.3.5 Robotic Interior Finishing

Interior finishing work is work conducted inside a structure (e.g., an office building) to complete construction of the space within the structure (e.g., painting and false ceiling and gypsum board installation). Due to the confined nature of interior finishing work traditional material handling equipment (such as cranes) cannot be used. The following four systems are examples of interior finishing robots:

- The Shimizu CFR 1 (Fig. 57.18a) could be manually and tele-operated and allowed the installation of suspended ceiling panels to a height of up to 3.5 m. The robot's compliant gripper mechanism was able to finely adjust the positions of the panels.
- The Tokyu ceiling panel installation robot (Fig. 57.18b) could position and adjust the panels and also fire nails into the panels to fasten them to the underlying ceiling system.
- Two variants (Fig. 57.18c,d) of interior finishing modular mobile robot, developed in Technical University of Munich.
- The Komatsu Mighty Hand LH (Fig. 57.18e) could position and adjust interior wall panels, glass panels, door frames, window casings and the outside walls (up to 350 kg) with high accuracy.

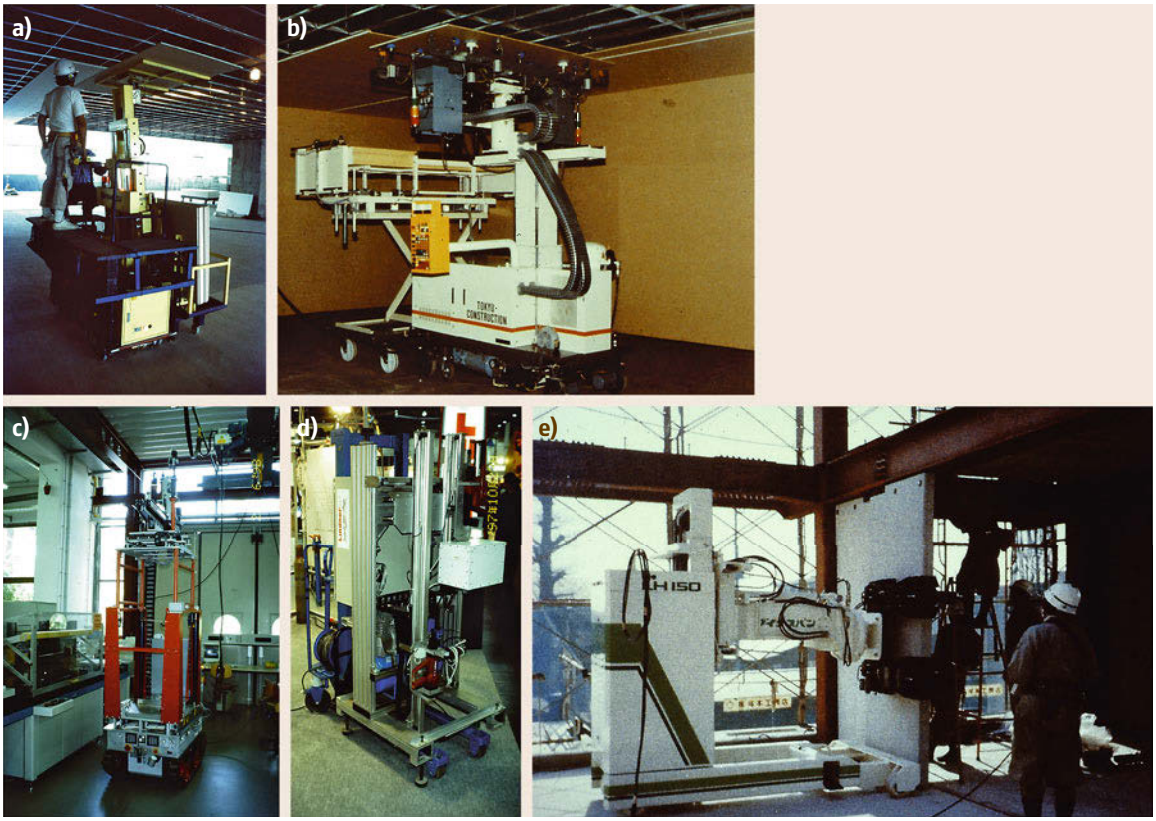


Fig. 57.18 **(a)** Shimizu CFR 1 – Interior finishing robot; **(b)** the Tokyu ceiling panel installation robot; **(c)** BR2 TU Munich – modular ceiling drilling robot system type 1; **(d)** BR2 TU Munich – modular ceiling drilling robot system type 2; **(e)** Kajima interior wall setting – KOMATSU LH 150

Concrete Finishing

Floor finishing is one of the most critical construction processes in which construction workers carry or guide trowels over an unfinished, wet concrete floor for several hours in a stooped posture. In order to reduce these physical demands on the workers and to ensure a more uniform finish quality, various companies developed and employed concepts for robots that can perform the concrete-leveling task: the Flat-Kun by Shimizu (800 m^2/h), the Kote King by Kajima (500 m^2/h), the Surfing Robo by Takenaka (300 m^2/h), the Obayashi made by Mitsubishi (500 m^2/h) and the Floor Traveling Robot MHE by Hazama (300 m^2/h). Each of these single task construction robots was able to operate on a floor where it was set up into any desired direction (e.g., not only move backwards and forwards and turning but also able to rotate 360° in place).

The finishing mechanism consisted, in most cases, of automatically controlled and operated rotating trowels. The degree of autonomy ranged from systems with human-machine interfaces for tele-operation to systems that could generate motions themselves and to preprogramming of paths for the robot to follow. In many cases gyroscopes and rotating laser levels assisted navigation and motion planning at a low level. After an intensive research and development phase that lasted until 1985, the first concrete finishing robots where deployed commercially to finish concrete floors in office buildings, factories, warehouses and shopping centers. The use of the robot systems became efficient once a floor plan allowed for a working area that could be processed without interruption of more than 500 to 600 m^2. Examples of commercial systems are depicted below in Fig. 57.19.

Tile Placement

The exterior of many buildings are sometimes finished with weather-resistant tiles. In Japan single-family buildings, factories, offices and high-rise buildings are often finished with tiles. Tiles are relatively small building elements compared to the total surface area of a building and thus a very large number of tiles have to be installed using the same, repetitive process, which involves applying mortar and positioning the tiles on

Avidbots Ex1016
Page 1528 of 2154

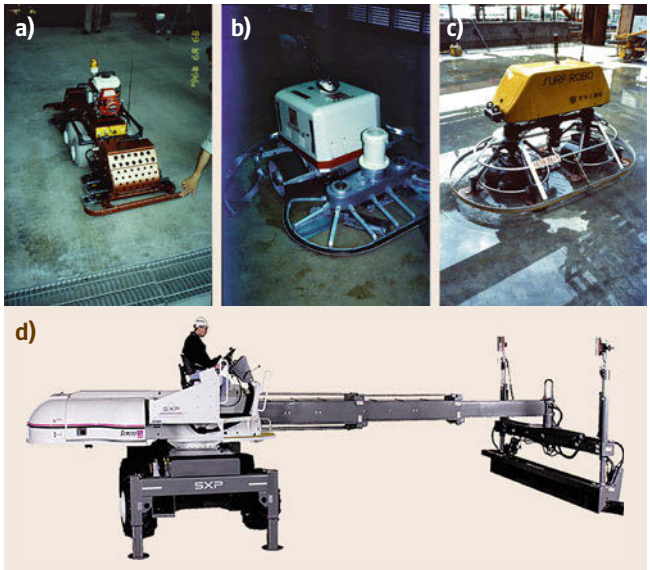


Fig. 57.19 **(a)** Hazama robot; **(b)** Kajima Kote King robot; **(c)** Takenaka Surf Robo; **(d)** auto-leveling concrete screed machine (courtesy Somero Enterprises, Inc.)

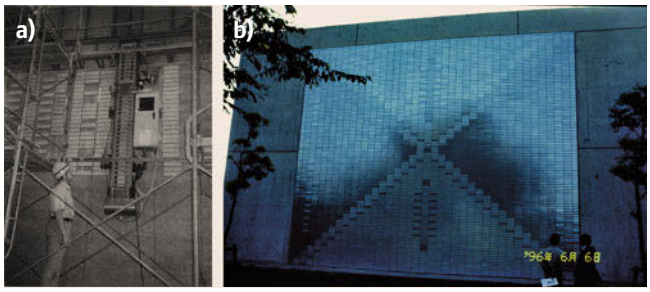


Fig.57.20a,b Hazama tile setting robot; **(a)** robot upon operation; **(b)** finished tile setting

top of the mortar. This and the fact that building facades are generally difficult to access make the use of automated systems feasible. Hazama's tile-setting robot (Fig. 57.20) also showed that accuracy could be enhanced and that the laying of patterns could be accomplished without dramatically increasing construction time.



Fig.57.21a,b Shimizu fireproof coating robots



Fig. 57.22 Concrete panel installation robot (courtesy of Fujita Research)

Fireproof Coating

In many countries building regulations require that steel structures be coated with fire proof materials and/or a fire-retarding paint. The application of the



Fig.57.23a–c Facade painting robots. **(a)** Shimizu; **(b)** Kajima; **(c)** Taisei

Avidbots Ex1016
Page 1529 of 2154

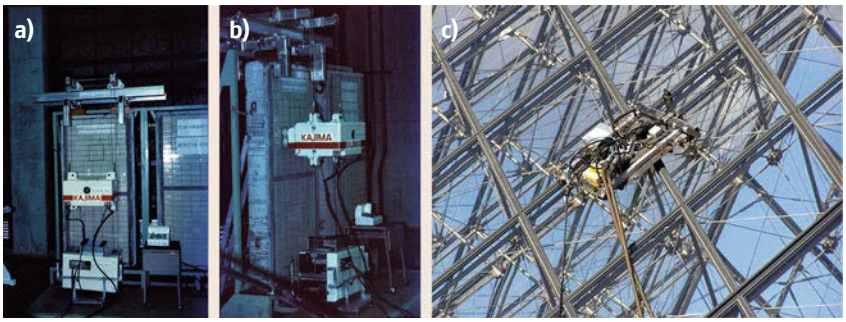


Fig.57.24a–c Kajima facade inspection robot **(a,b)**; Comatec Robosoft glass cleaning robot **(c)**

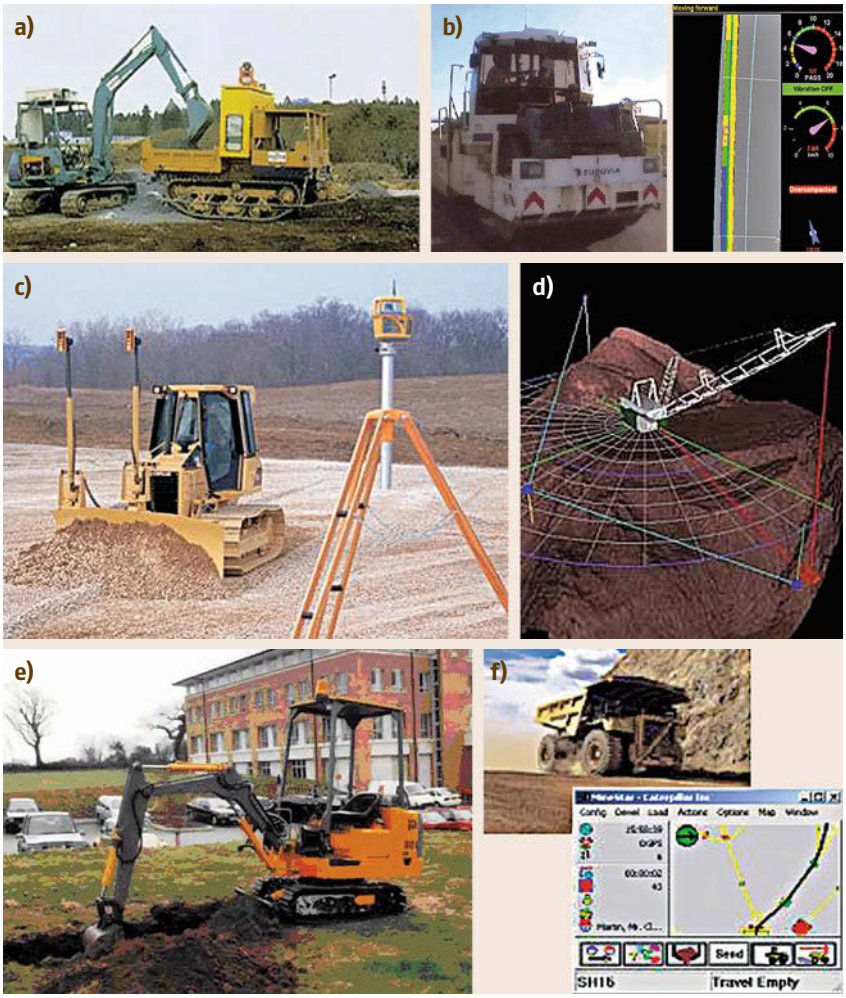


Fig.57.25a–f Earthmoving: **(a)** Teleoperated excavation system (courtesy Fujita Research); **(b)** computer assisted road compacting system (after [57.53]); **(c)** automated grading system (courtesy Caterpillar, Inc.); **(d)** automated drag line control system (after [57.54]); **(e)** autonomous excavator robot (after [57.55]); **(f)** autonomous off-road dump truck (after [57.56])

fire-roof coating and paint can only be done after the steel structure has been erected and joined on site (e.g., by welding) in order not to interfere with the joining process and in order to avoid any damage to the fire-proof coating. Therefore, it is not possible to shift the coating operations to upstream production steps that can be performed in structured factory environments where high efficiency can be achieved. Due to seismic considerations, most high-rise construction involves the use of steel structures and thus, the development and employment of automated and robotic systems that are able to apply the fire-proof coatings after the structure is erected took place in Japan. Robots devel-

oped by Shimizu (Fig. 57.21) and others were mostly autonomous.

57.3.6 Robotic Facade Operation

Facade operations involve the installation of windows, complete facade elements or building exterior walls. Facade elements are, in modern architecture and especially in high-rise construction, decoupled from the main load-bearing concrete or steel structure and can thus be considered as a type of *add-ons*. Facade installation operations are complex operations that involve the accurate positioning of heavy parts or elements at locations that are difficult to access (e.g., high elevations without scaffolding). This involves risk of injury (and thus extensive safety measures must be taken) and of damaging the building or the elements themselves. Furthermore, the positioning and alignment of prefabricated facade elements requires precision and low tolerances. A widespread trend (since the 1980s) of designing large buildings as monolithic structures that repeat the same or similar facade elements, has been a major motivation for investment in automated or robotic systems. Figure 57.22 shows an example of a tele-operated façade installation robot.

Painting

Facade-painting robots were developed to simplify the painting or re-painting of high-rise buildings during construction and operation. Facade painting robots have a particular advantage in that they can keep the paint finish quality constant. They usually have multiple spray nozzles and the spray area is either encapsulated or hermetically sealed in order to avoid streaking. A further advantage of facade painting robots is the fact that workers are not exposed to harmful paint fumes and vapours. Single task, facade-painting robots use one of the following three different strategies to move along the facade:

- Cable-suspended cage/gondola systems
- Rail guided systems
- Vacuum or other adhesion technology.

The use of façade-painting robots was not considered efficient for facades with an area $< 2000\,m^2$. Facade-painting robots were thus used primarily to paint large facades of warehouses and skyscrapers. Facades to be processed were required to have a low curvature and wherever possible no corners or lugs which can hinder the operation of the robot. Further, the design of window frames as well as the number of windows and the area they cover impact the feasibility and efficiency of facade-painting robots. Between 1984 and 1988 various companies introduced facade-painting robots. Shimizu and Kajima Corporations of Japan both applied the principle of the suspended cage or gondola (Fig. 57.23a,b, respectively). The Taisei Corporation of Japan used a rail-guided system and combined it with the gondola approach for their robot (Fig. 57.23c). The fastest of the systems (Kajima's robot) worked with a speed of $290\,m^2/h$ during the application of the primer paint, $200\,m^2/h$ for the base coat and $290\,m^2/h$ for the top coat.

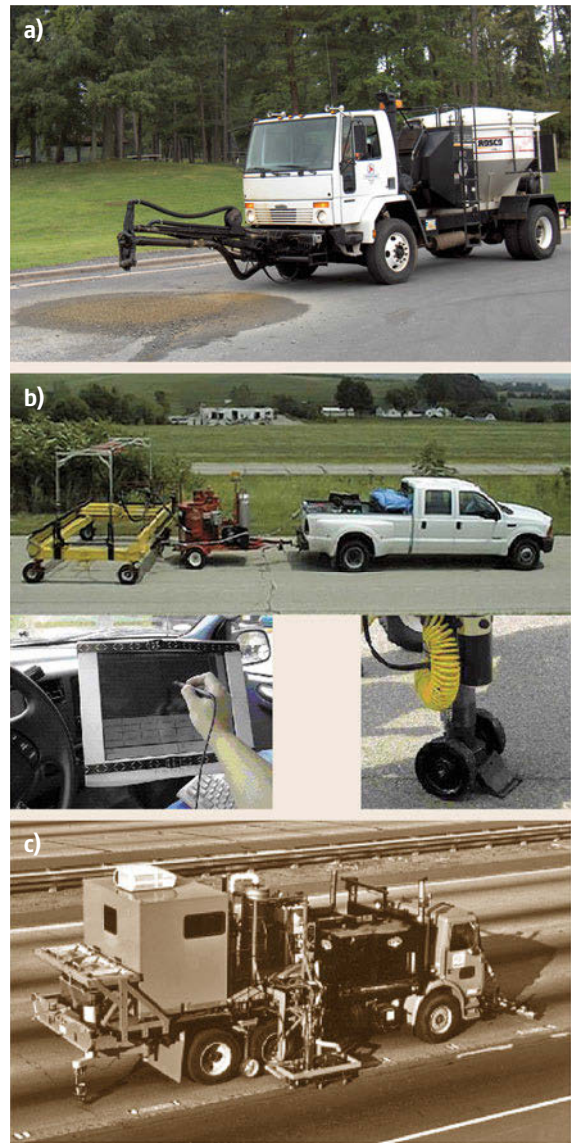


Fig.57.26a–c Road maintenance: **(a)** Teleoperated pothole patching robot (courtesy Leeboy); **(b)** semi-autonomous road crack sealing robot (after [57.57]); **(c)** semi-autonomous road crack sealing robot (after [57.58])



Fig.57.27a–c Material handling: **(a)** six degree of freedom robotic crane (after [57.59]); **(b)** large scale pipe manipulator; **(c)** large manipulator system (courtesy of Shimizu Corp.)

Robotized Inspection and Maintenance

Facades of high-rise buildings are in many cases equipped with tiles or other surface panels that have to be inspected regularly during the building's life-cycle in order to detect structural damage and in order to replace tiles or panels that might fall from the facade. Typically, workers access those tiles or panels via cages or gondolas suspended from the roof of the buildings. This work process was considered by Japanese construction firms as monotonous, inefficient and dangerous. Further, since the method for identifying damaged tiles or panels involves listening to the sound a tile or panel makes when gently impacted by a handheld tool, the sounds were difficult to classify at high elevations due to wind noise. Therefore, with substantial financial commitment, autonomous facade-inspection and maintenance robots were developed. Between 1985 and 1988, six different facade-inspection robots were developed in Japan by the Kajima, Takenaka, Obayashi, Taisei, Tamagawa and Seki Corporations. For the inspection of the facade of a 40 m high building (3000 m^2 facade) an inspection robot needed in average 8 h including ≈1 h for task as preparation, configuration, conversions, dismantling and cleaning of the robot (Fig. 57.24a,b).

Today robotic systems are not only developed for the construction of buildings but also for the operation, maintenance and decommissioning of buildings and other construction products. These types of *service* robots were developed for the construction sector for inspection of nuclear power plants, exterior walls of high-rise buildings and cleaning of high-rise facades or glass roofs (Fig. 57.24c).

57.3.7 Earthmoving

Earthmoving is a construction process for preparing project sites for construction by digging, grading, trenching, scraping and other similar tasks. There have been significant developments in many earthmoving areas and certain tasks (such as grading) can now be completely automated. Figure 57.25 shows several examples of automated earthmoving equipment some of which were still in the research and development phase while other are already commercial products.

57.3.8 Road Maintenance

Figure 57.26 shows three examples of machines that perform pothole repairs and crack sealing on road with various degrees of automation. Figure 57.26a is a commercial product while the other two machines were research projects undertaken by departments of transportation in the US.

57.3.9 Material Handling

The transport and handling of materials is a critical activity at most construction sites. Figure 57.27 shows examples of material handling robots developed under different research efforts.

57.4 Integrated Robotized Construction Sites

The first prototypes for mainly automated high-rise construction sites were put into operation in 1990 and 1991 by Shimizu (Fig. 57.28) after five years in development and a financial outlay of almost sixteen million Euros. Since then, twenty automated high-rise sites have been implemented by different companies (Taisei, Takenaka, Kajima, Maeda, and Kumagai) [57.60, 61].

57.4.1 Robotic Roof Field Factory Approach

An integrated robotized construction site involves the use of semi- and fully-automated storage, transport and assembly equipment and/or robots that are used to erect a building almost completely automatically. It is an attempt to improve the sequencing of construction

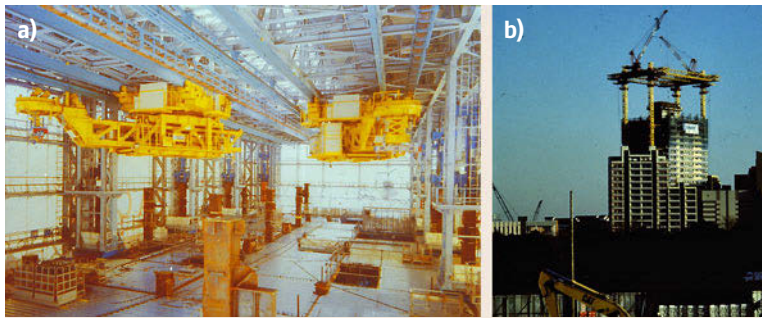


Fig. 57.28 (a) Robotic trolleys for transporting and positioning of beams, columns, floor panels, building services units and facades, in Shimizu SMART system; (b) SMART roof field factory view

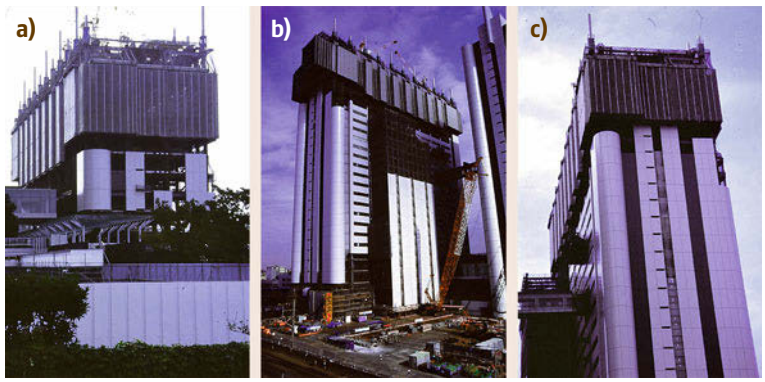


Fig.57.29a–c Obayashi ABCS (a) early construction phase; (b) intermediate construction phase; (c) final construction phase

processes and construction site management by using real-time control. This includes an unbroken flow of information from the planning and design phase through programming the on-site robots and controlling and monitoring the construction operations.

The robotic roof field factory approach is typically implemented once the building foundations have been laid. The production equipment, on which the steel construction has been installed with assembly and transport robots, is then covered completely with a roof made of plastic film. Depending on the roof system used, this process takes from three to six weeks after which the robots go into production. Due to the lack of space around building sites in Japan, steel and concrete plants are also often installed to supply parts in 10 min cycles on a just-in-time basis.



Fig.57.30a,b The AMURAD Construction Strategy: floor-wise upushing of the whole building. Kajima, Japan (a) early construction phase; (b) later construction phase

Prefabricated parts are checked and then placed in specific depots at the foot of the building or in the building itself where they are readily accessible by the robots. This is where the automated construction process actually starts. As many as 22 robots equipped with automatic crane winches deliver the columns, supports, floors, ceilings, walls and other elements to the floor of the steel skeleton under construction. These components are then positioned and fixed into place almost completely automatically. The steel columns and supports are joined together by welding robots after they have been positioned. The position and quality of the welding seams are monitored with lasers.

In the Obayashi ACBS (Automatic Constructions Building System) (Fig. 57.29 and in VIDEO 272), once a storey has been finished, the whole support structure, which rests on four columns, is pushed upwards by hydraulic presses to the next storey over a 1.5 h period. Fully extended, the support structure is 25 m high; retracted it measures 4.5 m. Once everything has been

moved up, work starts on the next storey. By constructing the topmost storey of the high-rise building as the roof at the beginning of the building process, the site is closed off in all directions, considerably reducing the effect of the weather and any damage it might cause.

57.4.2 Robotic Field Factory on-Ground Approach

The Amurad-system (Fig. 57.30), developed by the Kajima Corporation, is a way of construction based on the idea of after the first floor is built (which is the top floor), it is pushed up one floor at a time with the aid of large hydraulic cylinders. The plumbing, electrical and mechanical equipment and the interior fittings and cladding of the facade then begin. This process is repeated until the building is completed, where the last floor to be built becomes the ground floor. In order to realize the AMURAD System three automated systems were developed for pushing up the whole building, for transporting and assembling and for material handling.

57.4.3 Robotic Deconstruction and Robotic Recycling

Controlled deconstruction supported by robotic systems can be combined with component reuse systems since some elements such as structural steel have long life spans, and total recycling (i. e., scrapping, melting and recasting) consumes large amounts of energy. In a component reuse system all structural building components are accepted as trade-ins. Therefore, the deconstruction process comprises a reversed version of the construction process, which if it is based on industrialized fabrication, all deconstructed parts and elements could be directly reintroduced into the fabrication systems. The DARUMA system by Kajima Corporation (Fig. 57.31) enables such a controlled deconstruction process. Deconstructed structural steel elements are transported to special dismantling factories just-in-time

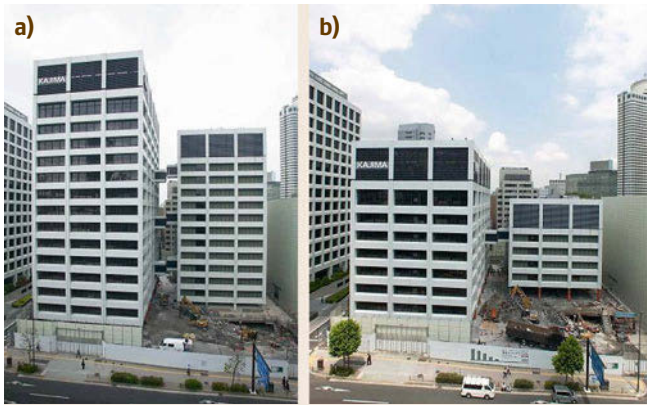


Fig.57.31a,b Systemic and automated floor-by-floor building deconstruction, Kajima, DARUMA System, Japan **(a)** early deconstruction phase; **(b)** later deconstruction phase

and just-in-sequence, where the old joints and finishes can be dismantled under factory conditions and fed into advanced reuse cycles. Modularized and standardized structural units can be inspected and renewed and then equipped with new finishes according to each customer's needs. *Component Reuse* systems could be connected to Advanced Construction ERP Systems and companies could match customers who want to sell their modular building for reuse and customers willing to buy reused building modules for further customization. Renewed building components are reorganized and customized in the factory and transported to other building sites. A combined system of controlled deconstruction and component reuse saves large amounts of materials and energy.

By adopting integrated industrialization processes the construction industry would have the chance to address all parameters relevant for sustainable economic, environmental and social development. Innovative industrialization-oriented architectural design structures, appropriate modularization and standardization of building structures, logistics, equipment and

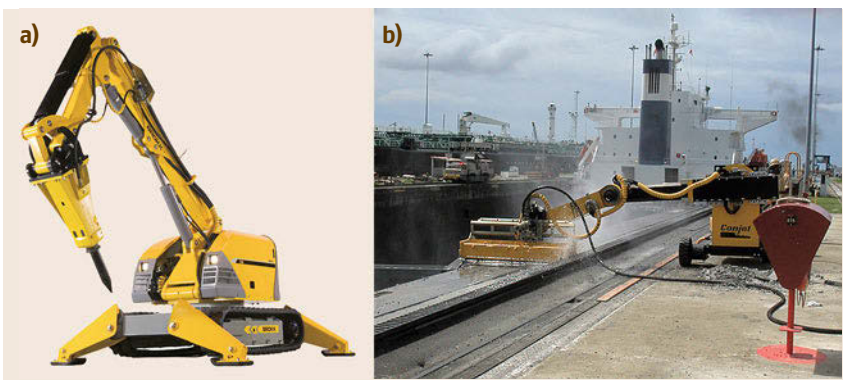


Fig. 57.32 **(a)** Teleoperated demolition robot (courtesy Brokk AB); **(b)** water-jet demolition robot (courtesy Conjet AB)

processes, could serve as a fundamental integration framework. Customized prefabrication could further be able to supply construction sites with individualized elements. Hyper-flexible robotic systems could support a limited amount of trained workers to perform positioning, joining and finishing operations. Construction ERP systems could support organization as well as a lean and demand oriented construction based on just-in-time and just-in-sequence resource supply meanwhile locally based factories grant identity and reduced logistic effort. Moreover integrated industrialization would not only be limited to the fabrication but also link systems of controlled deconstruction and component reuse to a network for continuous resource circulation.

In addition to fully automated deconstruction systems, a number of teleoperated demolition robots have been widely adopted in the construction industry. Two examples of these are shown in Fig. 57.32.

57.5 Currently Unsolved Technical Problems

A recent study found that the US construction industry does not realize approximately 15 billion US dollars per year in potential savings due to inadequate interoperability related to information exchange and management practices [57.62].

57.5.1 Interoperability

Although the lack of interoperability between the various information systems used in construction is a significant source of inefficiency for the industry, it is also a roadblock to the use of automated systems in construction. Automated systems need electronic information on past, current, and/or projected future states of a construction project to function efficiently.

For example, in order for a robotic crane to pick a steel beam from the site and deliver it to its target location, the robot must be able to know that the steel beam has been delivered to the site, as well as its current position and orientation. While information on the current inventory of parts on a site may be available on paper, it is rarely available electronically unless someone enters it manually into some computer system, which in turn may not be compatible with other systems used in that project. In many of the examples of automated construction technologies presented in this chapter, custom electronic databases and/or data formats have to be devised to demonstrate the robot's operation. In some cases, the electronic information had to be entered manually from paper into a computer for the robot to work correctly.

In addition to information exchange and management, many of the measurement instruments and sensors used in construction are not interoperable. This problem is not limited to the construction sector but is a relatively large problem in many robotic applications where different types of sensors are used. Several efforts are underway to make sensors interoperable [57.63] and construction equipment in general [57.64], but the issues have yet to be resolved.

57.5.2 Structural Connection Systems

Traditionally, structural member connections in construction have been designed for human installation. Whether using bolted, welded, or other types of connections, manual labor is usually involved in guiding the mating parts together and in establishing the connection.

For example, in structural steel erection, workers perched on the structure typically guide a crane operator through visual or auditory cues in order to maneuver a steel beam (or column) into place. The workers must then physically manipulate the beam in order to align corresponding surfaces for bolting or welding. Once the correct beam pose has been achieved, it must be maintained while the workers temporarily fasten the beam to the structure. The workers then release the beam from the crane and permanently fasten the beam to the structure at a later stage.

For automated (or robotic) construction to work, new connections that are more amenable to automation



Fig. 57.33 A drop-in, shear-load-only, steel connection

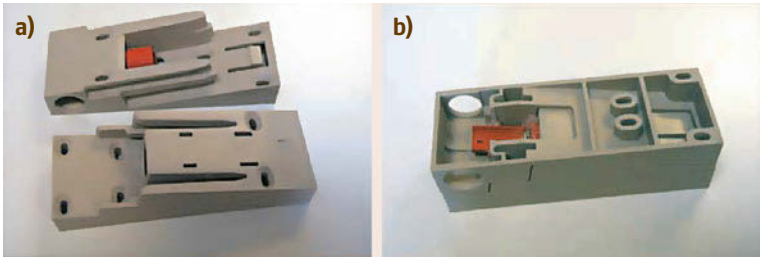


Fig.57.34a,b A piping and electrical drop-in connector shown **(a)** disassembled and **(b)** assembled

must be designed. These connections need not mimic traditional, human-installed connections, but should be optimized for use with robots instead. For example, the Lehigh University Advanced Technology for Large Structural Systems (ATLSS) Center designed a gravity-load-only shear steel connector [57.65] back in the early 1990s that is more suitable for automated construction (Fig. 57.33). This type of male–female connector for automatic assembly of building modules allows, together with an adequate control strategy, small tolerances, which permit assembly by automatic cranes [57.66].

An example of a nonstructural connector that is more amenable to automation is the piping and electrical connector developed for the FutureHome project, shown in Fig. 57.34 [57.21, 67].

Although automated welding has been applied in some limited form in construction, it is generally used to replace manual welding without changing the design of the parts being welded. In other words, no significant change in the way in which construction components are designed has occurred as a result of automated welding. Moreover, apart from a few examples of automated welding applied at the construction site, most of the other limited applications are done at the component fabrication facility.

57.5.3 Tolerances

Specifications for tolerances in the construction sector exist for most types of construction, however, they are not always achieved in practice [57.68]. For example, it has been stated that one of the biggest sources of problems in structural steel erection in the US is that fabricated pieces are often out of tolerance, and that this is only discovered during installation at the construction site [57.69]. However, since the finished facility must meet the design tolerances before it is accepted, this shifts the burden to workers and supervisors during construction. The workers are expected to handle problems as they come up rather than expect all fabricated construction components to be within tolerance. Since most construction projects are under tight schedules, it is often preferable to fix these kinds of problems on site rather than wait for replacement components to be fabricated and delivered.

However, tolerance problems are not all due to fabrication errors. Installation problems are also responsible for out-of-tolerance problems, for example, anchor rod installation has been an area of concern for structural steel erection. (Anchor rods (or bolts) are the connection interface between the concrete foundation and the structural steel columns. The rods are usually installed by the concrete foundation crew before the concrete is dry.) The situation of anchor rod patterns that do not match the hole patterns in a mating column is an identified problem in construction [57.26, 70].

Given the relatively loose achievable tolerances in construction, the application of robotics in construction faces an uphill battle. This, in addition to the unstructured nature of the construction site environment, requires that robots either be highly intelligent in order to correctly interpret and react to their surroundings or to be human assisted. However, site structure and tolerances are expected to improve as pressures to reduce costs and improve productivity continue to rise. Improvements in site organization and construction tolerances have already been proved to be achievable in a few cases, as has been demonstrated in Japan [57.71]. However, the economic case for these demonstration projects has yet to be made.

57.5.4 Power and Communications in the Field

Unlike manufacturing environments, in which specially designed factories are outfitted with the necessary power and communications installations, a construction project often begins before such resources have been installed at the site. Therefore, robots with large power requirements that need to communicate with supervisory systems located off site would be challenging to implement without significant added cost.

Although communications technologies have advanced significantly in the last few decades, it is still considered difficult to maintain a reliable local-area network at a construction site and to connect that network to the Internet. The use of cellular telephones

with a press-to-talk (the digital version of the traditional two-way radio communication method) feature has largely replaced the traditional two-way radios that also revolutionized on-site construction communications. However, in order for construction sites to become more automated, reliable interference-free high-bandwidth networks must be able to carry data transmission between sensors, machines, and supervisory systems.

57.5.5 Sensing

Field measurements are an integral part of the construction process. The *tape measure* and the *transit* have been used in construction for decades for measuring distances and angles, respectively. However, construction measurements are not limited to distances and angles, but can also include measurements of installed quantities, percentages of completion of activities, and so on. All of these measurements are necessary to be able to lay out the site where a facility is to be built, to measure the conformance of the as-built facility to the intended design, and to monitor safety, productivity, and progress.

In the past few decades, more advanced means of making measurements on construction sites have come into play. These include (but are not limited to) total stations, GPS, indoor GPS, ultra-wide band, laser scanners, ground-penetrating radar, equipment and structural health monitoring sensors, concrete maturity meters and radio frequency identification [57.72–84]. These technologies, as well as others that have not yet been adopted in the construction industry, are crucial for enabling more automation and robotics in construction. Nevertheless, despite all of these advances in technology, one of the biggest challenges for implementing robotics at the construction site is to be able to provide accurate and up-to-date information to the robots about where everything is and what other equipment are doing. This challenge will eventually be solved by the introduction of more sophisticated sensing and perception systems.

57.6 Future Directions

As previously discussed, developing new methods for collecting, processing, analyzing, and communicating construction information is a significant area of construction automation research, and will dominate near-term efforts. As this construction information becomes more readily accessible, automation of processes can be enabled directly from the combination of design information and current site status that is accurately captured and shared. Resource tracking will become ubiquitous and just-in-time delivery of needed materials and equipment will happen throughout the site.

Advances in mobility (humanoid robotics, smart cars, legged locomotion, etc.) will enable ever more automated material handling on the job site. These advances will require better control systems for the construction robots that can provide high payload and good positioning accuracy. The increased use of robots at the construction site will also drive research into safety systems for construction robots working around human workers and other machines.

Perhaps most important, more extensive automatic design systems will enable more prefabrication of building components and new methods of assembling those components on site, which in turn will provide the promise of faster, better, and cheaper construction robotics first envisioned in the 1980s.

57.7 Conclusions and Further Reading

The application of robotics to construction has yet to catch up with other industries such as automobile manufacturing. Construction presents a unique challenge for robotic applications. The construction environment is cluttered, unstructured, and teaming with human workers. In addition, construction processes are usually labor intensive and have to accommodate wide margins of error in the constructed facility. The application of robotics in construction to date has been limited to commercial teleoperated and programmable machines. Autonomous or semiautonomous machines are currently mostly limited to research projects within various nonconstruction organizations. With the increase in competition throughout the global construction market, construction companies are on the lookout for ways to improve productivity, quality, and safety. The use of automation and robotics is one answer that the industry is slowly turning toward. However, before these potential solutions can be successfully applied, much work is needed to improve construction tolerances, develop standards, and achieve real-time site status monitoring.

The International Association for Automation and Robotics in Construction holds an annual conference (the International Symposium on Automation and

Robotics in Construction) at which researchers can present the latest developments in the field. The proceedings from this conference hold a wealth of information on the state of the art of the field and are accessible to the general public through IAARC's website (www.iaarc.org).

There are several journals that publish articles on various aspects of automation in construction. Most notable of these are *Automation in Construction*, *Computer-Aided Civil and Infrastructure Engineering*, the *Journal of Computing in Civil Engineering*, and the *Journal of Construction Engineering and Management*. In addition, some journals publish special editions on robotics in construction. Some examples are *Autonomous Robots* and the *Journal of Advanced Robotic Systems*.

Video-References

VIDEO 272 Obayashi ACBS (Automatic Constructions Building System)
available from http://handbookofrobotics.org/view-chapter/57/videodetails/272

References

57.1 E. Ginzberg: The mechanization of work, Sci. Am. **247**(3), 66–75 (1982)

57.2 US Census Bureau: *Value of Construction Put in Place – Seasonally Adjusted Annual Rate* (U.S. Census Bureau, Washington DC 2012), http://www.census.gov/

57.3 Bureau of Labor Statistics: *Industries at a Glance: Construction: NAICS 23* (US Department of Labor, Washington DC 2012), http://www.bls.gov/iag/tgs/iag23.htm

57.4 US Census Bureau: *Statistics of U.S. Businesses* (U.S. Census Bureau, Washington DC 2012), http://www.census.gov/csd/susb/

57.5 European Commission: *Eurostat Regional Yearbook 2012* (Publications Office of the European Union, Luxembourg 2012), http://ec.europa.eu/eurostat/web/products-statistical-books/-/KS-HA-12-001

57.6 Statistics Bureau, Ministry of Internal Affairs and Communications: *Statistical Handbook of Japan* (Statistics Bureau, Tokyo 2012), http://www.stat.go.jp/english/data/handbook/index.htm

57.7 National Bureau of Statistics of China: *The Results of Preliminary Verified GDP for the First Three Quarters in 2012* (National Bureau of Statistics of China, Beijing 2012) http://www.stats.gov.cn/english/pressrelease/201211/t20121102_72217.html

57.8 National Bureau of Statistics of China: *China Statistical Yearbook* (China Statistics Press, Beijing 2012), http://www.stats.gov.cn/tjsj/ndsj/2012/indexeh.htm

57.9 D. Crosthwaite: The global construction market: A cross-sectional analysis, Constr. Manag. Econ. **18**(5), 619–627 (2000)

57.10 D.W. Halpin, R.W. Woodhead: *Construction Management*, 2nd edn. (Wiley, New York 1998)

57.11 K.S. Saidi: *Possible Applications of Handheld Computers to Quantity Surveying*, Dissertation (Univ. Texas, Austin 2002)

57.12 T. Greaves, B. Jenkins: *Capturing Existing Conditions with Terrestrial Laser Scanning: A Report on Opportunities, Challenges and Best Practices for Owners, Operators, Engineering/Construction Contractors and Surveyors of Built Assets and Civil Infrastructure* (Spar Point Research, Danvers 2004)

57.13 J.G. Everett, A.H. Slocum: Automation and robotics opportunities – Construction versus manufacturing, J. Constr. Eng. Manag. ASCE **120**(2), 443–451 (1994)

57.14 L.A. Demsetz: Task identification for construction automation, 6th Int. Symp. Autom. Robotics Constr. (1989) pp. 95–102

57.15 R. Kangari, D.W. Halpin: Potential robotics utilization in construction, J. Constr. Eng. Manag. **115**(1), 126–143 (1989)

57.16 R.L. Tucker: High payoff areas for automation applications, 6th Int. Symp. Autom. Robotics Constr. (1988) pp. 9–16

57.17 L. Cousineau, N. Miura: *Construction Robots: The Search for New Building Technology in Japan* (ASCE, Reston 1998)

57.18 J.G. Everett, H. Saito: Construction automation: Demands and satisfiers in the United States and Japan, J. Constr. Eng. Manag. ASCE **122**(2), 147–151 (1996)

57.19 M. Taylor, S. Wamuziri, I. Smith: Automated construction in Japan, Proc. ICE Civil Eng. **156**(1), 34–41 (2003)

57.20 J. Maeda: Current research and development and approach to future automated construction in Japan, Proc. Constr. Res. Congr. (2005) p. 2403

57.21 C. Balaguer, M. Abderrahim, J.M. Navarro, S. Boudjabeur, P. Aromaa, K. Kahkonen, S. Slavenburg, D. Seward, T. Bock, R. Wing, B. Atkin: FutureHome: An integrated construction automation approach, IEEE Robotics Autom. Mag. **9**(1), 55–66 (2002)

57.22 Y. Maruyama, Y. Iwase, K. Koga, J. Yagi, H. Takada, N. Sunaga, S. Nishigaki, T. Ito, K. Tamaki: Development of virtual and real-field construction management systems in innovative, intelligent field factory, Autom. Constr. **9**(5/6), 503–514 (2000)

57.23 C. Balaguer: Soft robotics concept in construction industry, World Autom. Congr. (2004) pp. 517–522
57.24 K.A. Reed: The role of the CIMSteel integration standards in automating the erection and surveying of structural steelwork, 19th Int. Symp. Autom. Robotics Constr. SP989 (NIST, Gaithersburg 2002)
57.25 N.J. Shih: The application of a 3-D scanner in the representation of building construction site, ISARC 2002: 19th Int. Symp. Autom. Robotics Constr. (2002) pp. 337–342
57.26 B. Akinci, F. Boukamp, C. Gordon, D. Huber, C. Lyons, K. Park: A formalism for utilization of sensor systems and integrated project models for active construction quality control, Autom. Constr. **15**(2), 124–138 (2006)
57.27 G.S. Cheok, W.C. Stone: Non-intrusive scanning technology for construction assessment, IAARC/IFAC/IEEE. Int. Symp. (1999) pp. 645–650
57.28 K. McKinney, M. Fischer: Generating, evaluating and visualizing construction schedules with CAD tools, Autom. Constr. **7**(6), 433–447 (1998)
57.29 B. Akinci, M. Fischer, J. Kunz: Automated generation of work spaces required by construction activities, J. Constr. Eng. Manag. ASCE **128**(4), 306–315 (2002)
57.30 V. Kamat, R. Lipman: Evaluation of standard product models for supporting automated erection of structural steelwork, Autom. Constr. **16**(2), 232–241 (2006)
57.31 C.M. Eastman: *Building Product Models* (CRC, Boca Raton 1999)
57.32 T. Bock, A. Malone: The Integrated Project ManuBuild of the EU, ISARC 2006 23rd Int. Symp. Autom. Robotics Constr. (2006) pp. 361–364
57.33 G. Aouad, J. Kirkham, P. Brandon, F. Brown, G. Cooper, S. Ford, R. Oxman, M. Sarshar, B. Young: Information modeling in the construction industry – The information engineering approach, Constr. Manag. Econ. **11**(5), 384–397 (1993)
57.34 G. Beer: Tunconstruct: A new european initiative, T&T Int. FEV (2006) pp. 21–23
57.35 H.M. Huang: *Autonomy Levels for Unmanned Systems (ALFUS) Framework Volume I: Terminology Version 2.0, NIST Special Publication 1011-I-2.0* (NIST, Gaithersburg 2008), http://www.nist.gov/el/isd/ks/upload/NISTSP_1011-I-2-0.pdf
57.36 Y.F. Ho, H. Masuda, H. Oda, L.W. Stark: Distributed control for tele-operations, IEEE/ASME Trans. Mechatron. **5**(2), 100–109 (2000)
57.37 S. Singh: State of the art in automation of earthmoving, ASCE J. Aerosp. Eng. **10**(4), 179–188 (2002)
57.38 H. Quang, M. Santos, N. Quang, D. Rye, H. Durrant-Whyte: Robotic excavation in construction automation, IEEE Robotics Autom. Mag. **9**(1), 20–28 (2002)
57.39 J. Albus, R. Bostelman, N. Dagalakis: The NIST RoboCrane, J. Robotic Syst. **10**(5), 709–724 (1993)
57.40 K.S. Saidi, A.M. Lytle, W.C. Stone, N.A. Scott: *Developments toward automated construction, NIST Interagency Rep. 7264* (NIST, Gaithersburg 2005)
57.41 S.C. Kang, E. Miranda: Physics based model for simulating the dynamics of tower cranes, 10th Int. Conf. Comput. Civil Build. Eng. (ICCCBE) (2004)
57.42 Weckenmann LLC: Machinery and plant systems for the production of precast concrete elements, http://www.weckenmann.com/en
57.43 M. Damlund, S. Goth, P. Hasle, K. Munk: Low back pain and early retirement among Danish semi-skilled construction workers, Scand. J. Work, Environ. Health **8**(1982), 100–104 (1982)
57.44 S. Schneider, P. Susi: Ergonomics and construction: A review of potential hazards in new construction, Am. Ind. Hyg. Assoc. J. **55**, 635–649 (1994)
57.45 T. Bock: *Robot Oriented Design* (Shokokusha Publishing, Tokyo 1988)
57.46 T. Bock: *A study on Robot-Oriented Construction and Building System, Thesis for Doctorate of Engineering, Report Number 108066* (University of Tokyo, Tokyo 1989)
57.47 T. Bock, T. Linner: *Robot-Oriented Design and Management* (Cambridge Univ. Press, Cambridge 2014)
57.48 T. Bock: The Japanese approach of SMAS-solid material assembly system and the European approach of ROCCO-robotic assembly system for computer integrated construction, EC-Japan Conf. (Reading University, Reading 1995)
57.49 G. Wickström, T. Niskanen, H. Riihimäki: Strain on the back in concrete reinforcement work, Br. J. Ind. Med. **42**(4), 233–239 (1985)
57.50 H. Benckert: Mechydronic for boom control on truck-mounted concrete pumps, Tech. Symp. Constr. Equip. Technol. 2003 (2003)
57.51 F. Gebhart, G. Mayer, F. Ott, A. Barren, B. Heid, W. Schencking, E. Andres Puente, T. Bock, A. Delchambre: Final report of the ROCCO project, ESPRIT III program of the European Union (1998)
57.52 T. Bock: Plenary paper: State of the art of automation and robotics in construction in Germany ROCCO: Robotic assembly system for computer integrated construction, 13th ISARC, Int. Conf. Autom. Robotics Constr., Tokio (1996)
57.53 F. Peyret: The Achievements of the computer integrated road construction project, 17th IAARC/CIB/IEEE/IFAC/IFR Int. Symp. Autom. Robotics Constr. (ISARC) (2000)
57.54 Commonwealth Scientific, Industrial Research Organisation: *Mining Robotics Project* (CSIRO, Clayton South 2006), https://wiki.csiro.au/display/ASL/Dragline+Automation
57.55 D.A. Bradley, D.W. Seward: The development, control and operation of an autonomous robotic excavator, J. Intell. Robotic Syst. **21**(1), 73–97 (1998)
57.56 P. Coal, C. Hughes: *Project C8001: Introduction of Autonomous Haul Trucks. Final Report* (Australian Coal Research, Brisbane 1997)
57.57 C. Haas, K. Saidi, Y. Cho, W. Fagerlund, H. Kim, Y. Kim: *Implementation of an Automated Road Maintenance Machine (ARMM), Center for Transportation Research, Project Summary Report* (NIST Interagency Rep., 7264 2005)
57.58 D.A. Bennett, X. Feng, S.A. Velinsky: AHMCT automated crack sealing program and the operator controlled crack sealing machine, Transp. Res. Board Annu. Meet. (2003)

Avidbots Ex1016
Page 1539 of 2154

57.59 A.M. Lytle, K.S. Saidi: NIST research in autonomous construction, Auton. Robots **22**(3), 211–221 (2007)
57.60 T. Linner: Automated and Robotic Construction: Integrated Automated Construction Sites, Dissertation (Universität München, München 2013)
57.61 T. Bock, T. Linner: *Logistics, Site Automation and Robotics: Automated/Robotic On-site Factories* (Cambridge Univ. Press, Cambridge 2014)
57.62 M.P. Gallaher, R.E. Chapman: *Cost Analysis of Inadequate Interoperability in the US Capital Facilities Industry* (National Institute of Standards and Technology, Gaithersburg 2004), US Dept. of Commerce, Technology Administration
57.63 K.B. Lee, M.E. Reichardt: Open standards for homeland security sensor networks, Instrum. Meas. Mag. IEEE **8**(5), 14–21 (2005)
57.64 E.F. Begley, M.E. Palmer, K.A. Reed: *Semantic Mapping Between IAI ifcXML and FIATECH AEX Models for Centrifugal Pumps* (National Institute of Standards and Technology, Gaithersburg 2005)
57.65 R. Fleischman, B.V. Viscomi, L.W. Lu: Development, analysis and experimentation of ATLSS connections for automated construction, Proc. 1st World Conf. Steel Struct. (1992)
57.66 S. Garrido, M. Abderrahim, A. Gimenez, C. Balaguer: Anti-swinging input shaping control of an automatic construction crane, IEEE Trans. Autom. Sci. Eng. **5**(3), 549–557 (2007)
57.67 T. Bock: Montage und Demontage im Holzbau mittels Schnellverschlüssen, BMBF Projektnummer: 0339835/5
57.68 J.K. Latta: *Inaccuracies in Construction*, Canadian Building Digest 171 (Institute for Construction, National Research Council Canada, Ottawa 1975), http://web.mit.edu/parmstr/Public/NRCan/CanBldgDigests/cbd171_e.html
57.69 A.M. Lytle, K.S. Saidi (Eds.): *Proceedings of the 23rd ISARC* (International Association for Automation and Robotics in Construction, Tokyo 2006)
57.70 A.M. Lytle, K.S. Saidi (Eds.): *Automated Steel Construction Workshop 2002* (National Institute of Standards and Technology, Gaithersburg 2004)
57.71 Y. Miyatake: SMART system: A full-scale implementation of computer integrated construction, 10th Int. Symp. Autom. Robotics Constr. (1993)
57.72 C. Lindfors, P. Chang, W. Stone: Survey of construction metrology options for AEC industry, J. Aerosp. Eng. **12**, 58 (1999)
57.73 S. Kang, D. Tesar: A novel 6-DoF measurement tool with indoor GPS for metrology and calibration of modular reconfigurable robots, IEEE ICM Int. Conf. Mechatron., Istanbul (2004)
57.74 L.E. Bernold, L. Venkatesan, S. Suvarna: Equipment mounted multi-sensory system to locate pipes, Pipelines **130**, 112 (2004)
57.75 D.A. Willett, K.C. Mahboub, B. Rister: Accuracy of ground-penetrating radar for pavement-layer thickness analysis, J. Transp. Eng. **132**, 96–103 (2006)
57.76 C.L. Barnes, J.F. Trottier: Effectiveness of ground penetrating radar in predicting deck repair quantities, J. Infrastruct. Syst. **10**, 69 (2004)
57.77 J.A. Huisman, S.S. Hubbard, J.D. Redman, A.P. Annan: Measuring soil water content with ground penetrating radar – A review, Vadose Zone J. **2**(4), 476–491 (2003)
57.78 G.W. Housner, L.A. Bergman, T.K. Caughey, A.G. Chassiakos, R.O. Claus, S.F. Masri, R.E. Skelton, T.T. Soong, B.F. Spencer, J.T.P. Yao: Structural control: Past, present, and future, J. Eng. Mech. **123**(9), 897–971 (1997)
57.79 US Department of Transportation: *Maturity Meters: A Concrete Success*, ed. by L. Pope (Federal Highway Administration (FHWA), Washington 2002)
57.80 S.V. Ramaiah, B.F. McCullough, T. Dossey: *Estimating in situ Strength of Concrete Pavements Under Various Field Conditions* (Univ. of Texas, Austin 2001), Center Transport. Res.
57.81 J. Song, C. Haas, C. Caldas, E. Ergen, B. Akinci, C.R. Wood, J. Wadephul: *Field Trials of RFID Technology for Tracking Fabricated Pipe – Phase II* (FIATECH, Austin 2003), http://www.fiatech.org/images/stories/techprojects/project_deliverables/SC_FieldTrialsofRFIDTechnologyforTrackingFabricatedPipe_PhaseII.pdf
57.82 J. Song, C.T. Haas, C. Caldas, E. Ergen, B. Akinci: Automating the task of tracking the delivery and receipt of fabricated pipe spools in industrial projects, Autom. Constr. **15**(2), 166–177 (2006)
57.83 J. Aksoy, I. Chan, K. Guidry, J. Jones, C. R. Wood: *Materials and Asset Tracking Using RFID: A Preparatory Field Pilot Study* (FIATECH, Austin 2004), http://www.fiatech.org
57.84 J. Kang, P. Woods, J. Nam, C.R. Wood: *Field Tests of RFID Technology for Construction Tool Management* (FIATECH, Austin 2005), http://www.fiatech.org

Avidbots Ex1016
Page 1540 of 2154